19941228 105

INVESTIGATION INTO THE BEHAVIOR OF GEOMETRICALLY
NONLINEAR COMPOSITE ARCHES

THESIS

John C. Bailey
Captain, USAF

AFIT/GAE/ENY/94D-02

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/94D-02

INVESTIGATION INTO THE BEHAVIOR OF GEOMETRICALLY
NONLINEAR COMPOSITE ARCHES

THESIS

John C. Bailey
Captain, USAF

AFIT/GAE/ENY/94D-02

Approved for public release; distribution unlimited

The views expressed in this thesis are those of

the author and do not reflect the official

policy or position of the Department of Defense

or the U. S. Government

INVESTIGATION INTO THE BEHAVIOR OF GEOMETRICALLY

NONLINEAR COMPOSITE ARCHES

THESIS

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Aeronautical Engineering

John C. Bailey

Captain, USAF

December 1994

Acknowledgments

Although only one name is printed on the title page of this thesis, it is by no means the work of just one person. As such, I must thank Dr Anthony Palazotto for pounding the concepts related to nonlinear shell theory into my brain despite great unconscious resistance. I would also like to thank Dr Arnold Mayer (WL/FIV) and Dr Arje Nachman (AFOSR) for sponsoring my efforts. Most importantly, I need to thank my wife Camille and my son Tyler, because these times were no easier on them than they were on me.

Table of Contents

List of Figures

List of Tables

ABSTRACT

This research modifies the existing finite element formulation of a potential energy based large deformation and moderate rotation theory. Hermitian shape functions replace the existing linear bending angle interpolations. Negligible differences between the two formulations indicate the underlying kinematics limit the accuracy, not the finite element interpolations. Using the new program, numerous nonlinear arch geometries are modeled to investigate the effects of arc length and thickness variations. Local and global snapping phenomena are captured as well as through the thickness shear driven nonlinearities.

INVESTIGATION INTO THE BEHAVIOR OF GEOMETRICALLY
NONLINEAR COMPOSITE ARCHES

I.   Introduction

1.1 Background

Composite shells, plates, arches, and beams are the building blocks for the majority of today's complex aerospace structures.   The critical nature of such structures requires accurate prediction of their behavior under any loading conditions.   However, analysis beyond the basic linear geometric and material regions is complex and time consuming.

One type of complex shell analysis combines small strains with large displacements and rotations.   In this situation, the material remains linearly elastic while the geometry becomes non-linear. Geometric non-linearity causes dramatic structural stiffness changes without significant material changes.   Palazotto and Dennis present the Simplified Large Displacement/Rotation (SLR) theory and complementary FORTRAN code to analyze shells experiencing these loading conditions (19).   SLR theory determines the equilibrium path of orthotropic shells using a total Lagrangian approach and includes a parabolic transverse shear stress distribution.   Equations are solved using the finite element method.   The result is a FORTRAN code capable of analyzing linearly elastic behavior of isotropic and laminated orthotropic cylindrical shell structures undergoing geometric non-linearity with large deformations and moderate rotations.

Creaghan reduced SLR theory to one dimension for the analysis of beams and arches (3; 4).   The resulting FORTRAN code combines all the

features of Palazotto and Dennis's work with extensible mid-plane strain terms from Smith (25; 26). Creaghan also include displacement control and Riks methods for solving non-linear equations. These modifications allow uninhibited observation of SLR displacement and rotation limits. Creaghan's theory produces accurate solutions through 23 degrees of rotation.

Miller improved Creaghan's rotation limits to approximately 45 degrees (13). Miller replaced Creaghan's small angle approximation for the bending angle (ψ) with a truncated series representation of the tangent function, thereby introducing large rotation kinematic theory to Creaghan's formulation.

The purpose of this thesis is to improve the rotation limit of Miller's code. Miller's adaptation of the finite element method (FEM) will be altered to include higher order shape functions for ψ. Refining the finite element model will improve solutions within the limits of Miller's theory. The resulting FORTRAN code will then be used to analyze arch classification parameters.

1.2 Preview

Thickness and depth primarily determine the behavior of shells and arches. For example, shear plays a large role in thick structures but is negligible in thin shells; therefore, a theory which considers transverse shear should be used for analysis. Also, it is important to note a shallow shell/arch behaves very much like a plate/beam whereas a deep shell/arch does not. Obviously, the geometry of the structure should be classified before analysis begins.

Huang proposes thin arches have a very small physical thickness (h) vs. radius of curvature (r) ratio ($h/r \ll 1$) (10). Miller also

considers the half chord length (*c*), but his decision value ($c/h < 25$) is arbitrary (13).

Smith proposes a shallow shell has a small depth (δ) vs. half chord length ratio ($\delta/c < 0.3$) (25). Fung and Kaplan propose $\lambda = 2[3(1-\upsilon^2)]^{.25}(\delta/h)^{.5}$ as a classification criteria, limited to small opening angle ($\alpha << 1$) (8). Kaplan revises this definition to remove the opening angle limitation resulting in $\lambda = [12(1-\upsilon^2)]^{.25}(r/h)^{.5}(\alpha/2)$ (12). Miller uses $\lambda > 8$ to define deep shells (13).

The complex nature of large deformation and rotation theory requires solving numerous simultaneous, non-linear, differential equations. Even with simplifying assumptions, these equations remain complex and extensive. For this reason, most theories employ FEM for solution.

Accurate FEM solutions require proper shape functions. Cook, Gallagher, and Zienkiewicz all discuss shape functions at great length (2; 9; 29). Shape functions interpolate quantities within an element and are generally classified as C^0 or C^1 continuous. All shape functions must be C^0 continuous; that is, the quantity they interpolate must be continuous at each node. If a quantity is represented by C^1 shape functions, the first derivative of the quantity is continuous as well. Obviously, C^1 or Hermitian shape functions are required if a quantity and its first derivative must have interelement continuity. Miller uses linear C^0 shape functions to estimate ψ (13). As such, the first derivative of ψ ($\psi_{,2}$) is constant within a given element and discontinuous at each node. The current work will employ both quadratic C^0 and Hermitian interpolations of ψ.

1.3 Literature Review

Although primarily concerned with non-linear shell theory, this thesis also addresses arch classification parameters and finite element method shape functions. Therefore, a brief review of these topics is in order.

Kirchhoff's theory is one of the first for analysis of thin plates (22). His assumptions allow two dimensional analysis of three dimensional plate behavior. He assumed an inextensible mid plane and negligible normal strains. Kirchhoff also eliminated transverse shear effects by disallowing cross sectional warping.

Love took the next logical step in the development of shell theory (22). He adapted Kirchhoff's plate theory to curved shells. The Kirchhoff-Love theory obviously neglects transverse shear strain effects. As a result, both Kirchhoff's theory and Love's revision only apply to thin, isotropic plates experiencing small deflections.

Reissner and Mindlin improved Kirchhoff-Love theory through the addition of transverse shear considerations (14; 21). Reissner-Mindlin (RM) theory assumes constant through the thickness shear and, as a result, cross sections normal to the datum surface rotate but do not warp. This first order theory can cause shear locking in a finite element formulation because boundary conditions are not satisfied on the top and bottom of the shell. Shear correction factors minimize this condition.

Reddy adds a parabolic transverse shear strain representation to RM theory to eliminate shear locking (20). Palazotto and Dennis, Creaghan, and Miller employ Reddy's theory (3; 13; 19). As such, it is retained here.

Beam and arch theory spawn an enormous amount of research. As with shell theory, arch theory refinements are directed towards improving displacement and rotation limits. As a baseline for comparison, this thesis retains Miller's theory without changes (13). His theory combines total Lagrangian kinematics with the aforementioned trigonometric representation of the bending angle and vector decomposition of the displacement equations. Miller also includes parabolic transverse shear, mid-plane extensibility, and all non-linear in-plane strain terms. A finite element formulation approximates solutions.

Oliver and Oñate present a total Lagrangian finite element formulation which accurately calculates large displacements and rotations (18). They include a parabolic shear representation and mid plane extensibility. Degenerated shell elements similar to SLR theory are used as well as exact trigonometric relations. This is a stress based theory whereas SLR is strain based.

Surana also uses a vector decomposition of the displacement equations (27). Total Lagrangian kinematics and a finite element representation are employed as well.

Noguchi and Hisada use a finite element model to analyze post buckling shell response (17). Their vector based, total Lagrangian kinematics provide a good comparison to the current derivation.

Mondkar and Powell develop equations of motion based on total Lagrangian kinematics (16). They retain higher order terms and consider mid-plane extension and transverse shear. · Solutions for isotropic structures are found using FEM.

Huddleston uses a total Lagrangian formulation with no small angle assumptions to analyze deep, isotropic, circular arches (11).

Transverse shear is neglected while mid plane extension is allowed. Huddleston solves simultaneous, non-linear, first-order differential equations rather than using finite element or finite difference methods.

Sabir and Lock also neglect transverse shear and eliminate some higher order mid-plane strain terms (23). Total Lagrangian kinematics are used in conjunction with a finite element solution method to analyze isotropic, circular arches.

DeDeppo and Schmidt present numerous approaches to solving non-linear circular arch problems (5; 6; 24),. When they consider transverse shear, a first order representation is used. When mid plane stretching is considered, higher order terms are neglected. They use total Lagrangian kinematics with no small angle approximations and finite difference solution method.

Epstein and Murray retain quadratic in-plane strain terms while neglecting transverse shear strain (7). Mid-plane extension is considered with higher order terms retained. Epstein and Murray use finite elements and a total Lagrangian approach to examine only isotropic beams.

Belytschko and Glaum use small angle approximations in an updated Lagrangian basis (1). Mid-plane stretch is allowed but higher order terms are eliminated. Also, they ignore transverse shear.

Minguet and Dugundji also use the co-rotational method; however, Euler angles are used for exact rigid body kinematics (15). Transverse shear and mid-plane extensibility are considered without higher order terms.

II. Theory

2.1 Overview

The current work is based on Miller's kinematic development, constitutive relations, strain relations, and beam energy considerations (13). His work is reviewed here for completeness and to highlight certain inaccuracies in the theory. Miller's FEM formulation is modified to include a higher order shape function for the bending angle. Numerical solution methods and arch geometry definitions are discussed as well.

2.2 Kinematics

Miller derives arch displacement equations in a vector format. The first step considers displacements due to pure bending as shown in Figure 2-1. Points j and k are on the outer surface of the beam and
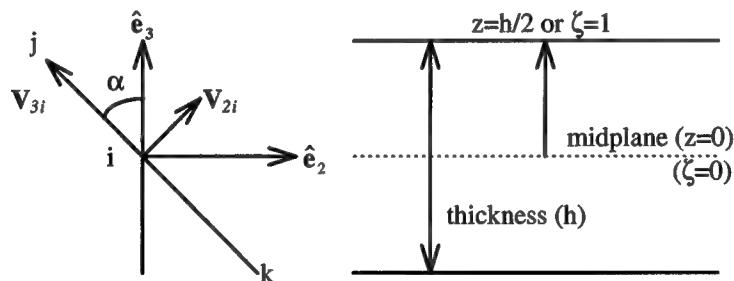
Figure 2-1 Flat Beam Bending Rotation

point i is on the mid plane. The V coordinates are initially parallel to the original system and move with the normal of the cross section during deformation. The $\hat{e}$ coordinates are the global coordinates of the total Lagrangian system. The vector V_{3i} remains parallel to the cross

sectional thickness during bending. The magnitude of V_{3i} equals the beam thickness (h). V_{3i} remains perpendicular to V_{2i} during pure bending and α is the rotation angle about V_{1i}. The natural coordinate ζ is aligned in the $\hat{e}_3$ direction with magnitude 1 at the top of the beam (positive $\hat{e}_3$ direction) and -1 at the bottom. V_{3i} is expressed in global coordinates as:

$$V_{3i} = (y_i - y_k)e_2 - (z_j - z_k)e_3 \qquad (2\text{-}1)$$

with direction cosines:

$$\begin{bmatrix} m_{3i} \\ n_{3i} \end{bmatrix} = \frac{1}{h}\begin{bmatrix} y_j - y_k \\ z_j - z_k \end{bmatrix} \qquad (2\text{-}2)$$

Eqn(2-2) is substituted into Eqn(2-1), yielding:

$$V_{3i}^n = h(m_{3i}^n e_2 + n_{3i}^n e_3) \qquad (2\text{-}3)$$

The n superscript denotes any deformed state. The position of any point P on the cross section is described by its location from the beam center line and the location of point i on the mid plane as

$$\begin{bmatrix} y \\ z \end{bmatrix}^n = \begin{bmatrix} y_i \\ z_i \end{bmatrix} + \frac{\zeta}{2} V_{3i}^n \qquad (2\text{-}4)$$

The total Lagrangian displacement of any point P is:

$$\overline{u}(z) = \begin{bmatrix} y_i \\ z_i \end{bmatrix}^n - \begin{bmatrix} y_i \\ z_i \end{bmatrix}^0 + \frac{\zeta}{2}[V_{3i}^n - V_{3i}^0] \qquad (2\text{-}5)$$

The superscript 0 denotes the original, undeformed state. Miller assumes vertical displacement (w) is constant through the thickness. As shown in Figure 2-2, P only displaces horizontally due to bending. This assumption implies stretching of the beam normal. Resulting normal strains are neglected. Horizontal displacement is assumed in order to keep w constant through the thickness. The resulting strains become
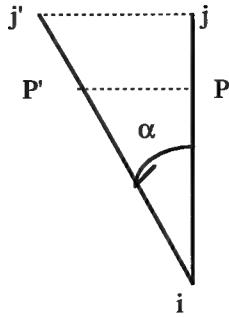
Figure 2-2 Bending Motion

significant as α approaches $\pi/2$. This assumption changes Eqn(2-3) as follows:

$$V_{3i}^n = \frac{h}{\cos(\alpha_i)}\begin{bmatrix} m_{3i} \\ n_{3i} \end{bmatrix} \qquad (2-6)$$

where

$$m_{3i} = -\sin(\alpha_i)$$
$$n_{3i} = \cos(\alpha_i)$$

Eqn(2-6) is substituted into Eqn(2-5) at $\alpha_0 = 0$ yielding:

$$\bar{u} = \bar{u}_i + \frac{\zeta}{2} h \left[\frac{1}{\cos(\alpha_i)} \begin{bmatrix} m_{3i} \\ n_{3i} \end{bmatrix}^n - \begin{bmatrix} m_{3i} \\ n_{3i} \end{bmatrix}^0 \right]$$

so that   (2-7)

$$V_{3i}^n = \begin{bmatrix} 1 & -\tan(\alpha_i) \\ 0 & 1 \end{bmatrix} V_{3i}^0 = [R]V_{3i}^0$$

where $\bar{u}_i$ represents the motion of point P due to the motion of point i on the midplane and [R] is the bending rotation tensor. The general displacement equation using natural coordinates is

$$\bar{u} = \bar{u}_i + \frac{\zeta}{2}[[R]-[I]]V_{3i}^0 \qquad (2-8)$$

where [I] is the 2X2 identity matrix.

Adapting this derivation to curved beams requires consideration of the additional motion due to curvature. Orthogonal curvilinear coordinates are used for arches as shown in figure 2-3. The s direction
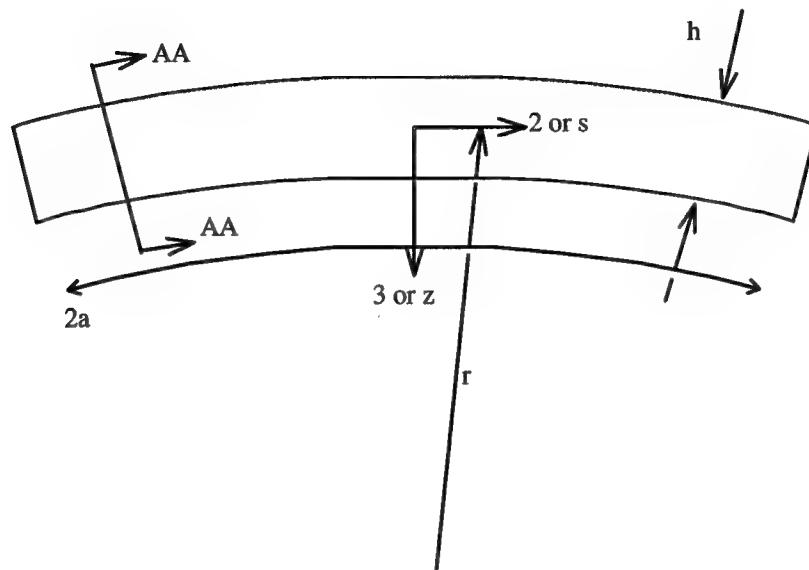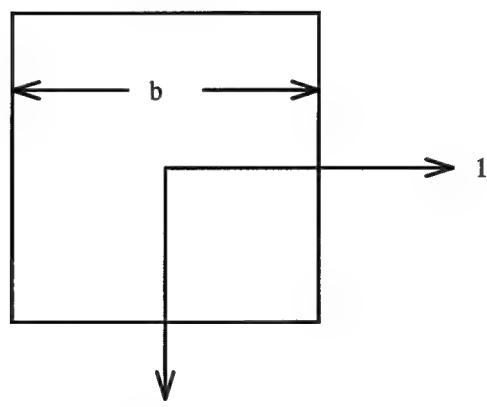
Figure 2-3 Global Arch Coordinates

runs along the arch midplane and the 3 direction points toward the center of curvature.

The $\bar{u}_i$ term in Eqn 2-8 represents the displacement of an arbitrary

point caused by non-bending motion. For straight beams, mid-plane

motion directly correlates to $\bar{u}_i$ for any arbitrary point. As shown in

figure 2-4, the $\bar{u}_i$ becomes thickness dependent in a curvlinear system

therefore requiring metric tensor coefficients (19; 22). In this case,
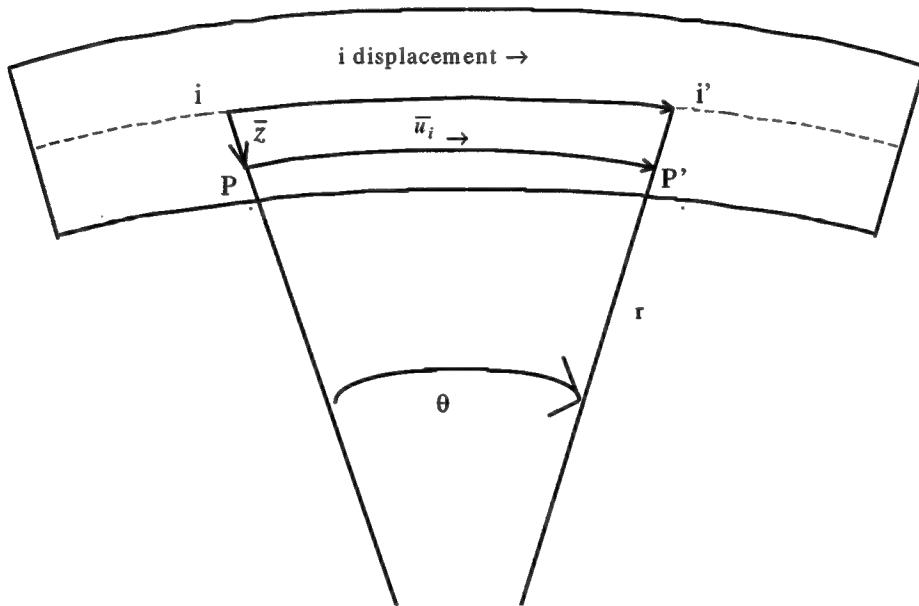
Figure 2-4 Mid Plane Displacement Effects

the motion of point P due to midplane motion is:

$$\bar{u}_i = \begin{bmatrix} v(1-\dfrac{z}{r}) \\ w \end{bmatrix}$$ (2-9)

where v and w are the motion of point i on the midplane. Using the

shell scale factors as did Palazotto and Dennis yields similar results

(19).

Figure 2-5 shows the relationship between ψ and slope ($w_{,2}$). A

positive ψ is defined as the angle which causes a point with a positive
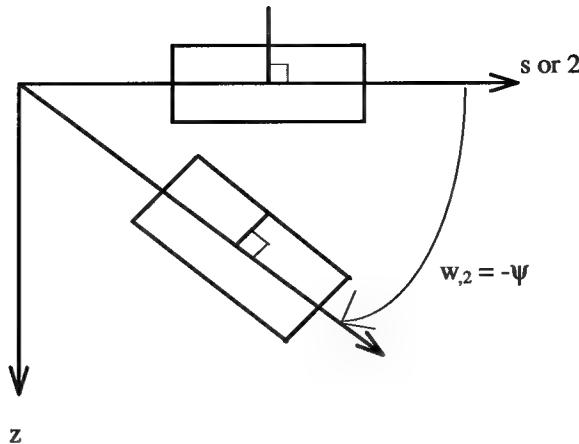
Figure 2-5 Pure Bending Convention

z coordinate to move in the positive s direction.   Since α is defined using the right hand rule in a positive direction about the 1 axis, $\psi = -\alpha$.

Expressing Eqn(2-8) and Eqn(2-9) in global coordinates yields these displacement components:

$$u_2 = v\left(1-\frac{z}{r}\right)+z\tan(\psi)$$

$$u_3 = w$$

(2-10)

These are the displacement relationships for bending only.   Instead of using a small angle approximation for $\tan(\psi)$, Miller used shell displacement relationships developed by Palazotto and Dennis to retain the tangent function in the u_2 component(13; 19).   As a result, the in plane displacement of Eqn(2-10) becomes:

$$u_2(s,z) = v\left(1-\frac{z}{r}\right)+z\tan(\psi)+z^3 k(\psi + w_{,2})$$

where:

(2-11)

$$k = \frac{4}{3h^2}$$

Similar to ψ, a positive shear angle β causes a point with a positive z coordinate to move in the positive s direction. Figure 2-6 shows the relationship between β and $w_{,2}$. Mathematically, $w_{,2}$, β, and ψ are related by:

$$w_{,2} + \psi = -\beta \qquad (2\text{-}12)$$

Eqn(2-12) is used to find β after $w_{,2}$ and ψ are found using FEM. The sign convention of these three quantities is shown in figure 2-7.
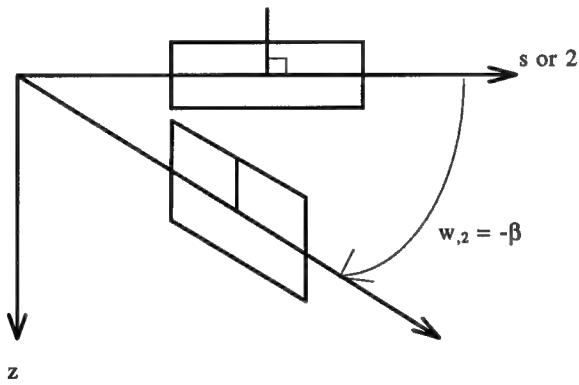
Figure 2-6 Angular Shear convention

The tangent function in Eqn(2-11) is approximated using the following series expansion:

$$\tan(\psi) = \sum_{i=0}^{\infty} -1^{i-1} \frac{2^{2i}(2^{2i}-1)}{2i!} B_{2i} \psi^{2i-1} \qquad (2\text{-}13)$$

where the B_n's are Bernoulli numbers and ψ is in radians (12). This series expansion is truncated at two terms yielding:

$$\tan(\psi) = \psi + \frac{1}{3}\psi^3 \qquad (2\text{-}14)$$
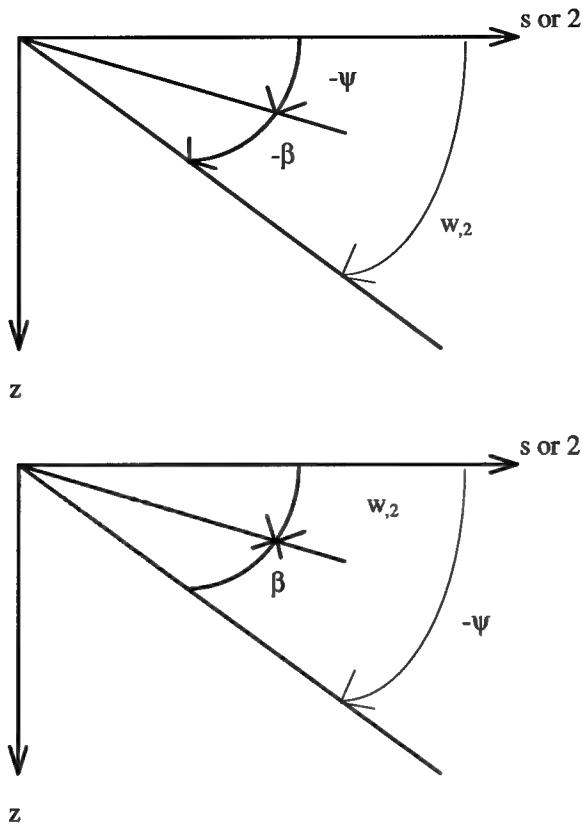
Figure 2-7 Slope, Shear, and Bending Relationship

Substituting Eqn(2-14) into Eqn(2-11) for u_2, one obtains the final displacement components:

$$u_1 = 0$$
$$u_2(s,z) = v\left(1-\frac{z}{r}\right) + z(\psi + \frac{1}{3}\psi^3) + z^3 k(\psi + w_{,2}) \qquad (2\text{-}15)$$
$$u_3(s) = w$$

where v and w are midplane displacements in the 2 and 3 directions, respectively, z is the distance from the midplane, ψ is angle due to bending, and $w_{,2}$ is the slope of the midplane tangent.

2-8

2.3 Constitutive Relations

Table 2.1 presents the explicit and contracted notation used for both the stress and strain tensors.

A fiber composite lamina has the local coordinate system defined as shown in figure 2-8. The 1' direction is aligned with the fibers of

Table 2-1 Contraction Definitions

Stress		Strain	
Contracted	Explicit	Contracted	Explicit
σ_1	σ_{11}	ε_1	ε_{11}
σ_2	σ_{22}	ε_2	ε_{22}
σ_3	σ_{33}	ε_3	ε_{33}
σ_4	σ_{23}	ε_4	$2\varepsilon_{23}$
σ_5	σ_{13}	ε_5	$2\varepsilon_{13}$
σ_6	σ_{12}	ε_6	$2\varepsilon_{12}$
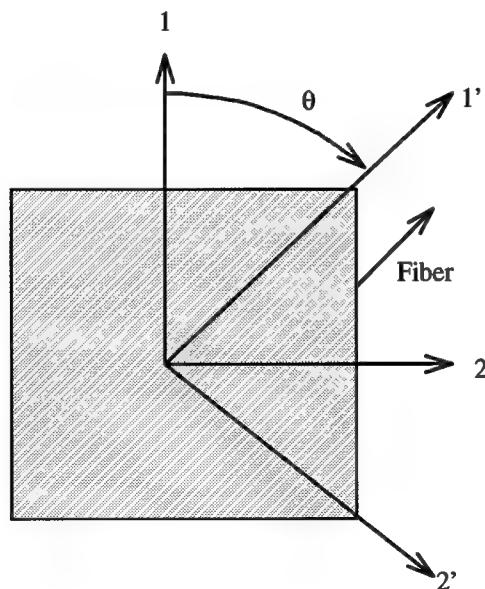
Figure 2-8 Material vs. Global Coordinates

the composite. The 2' and 3' directions are the transverse direction.

In this work, the 3 and 3' directions are aligned and point into the

page.

Following these conventions, the stress and strain for a composite

lamina are:

$$\begin{Bmatrix} \sigma'_1 \\ \sigma'_2 \\ \sigma'_6 \\ \sigma'_4 \\ \sigma'_5 \end{Bmatrix} = \begin{bmatrix} Q_{11} & Q_{12} & 0 & 0 & 0 \\ Q_{12} & Q_{22} & 0 & 0 & 0 \\ 0 & 0 & Q_{66} & 0 & 0 \\ 0 & 0 & 0 & Q_{44} & 0 \\ 0 & 0 & 0 & 0 & Q_{55} \end{bmatrix} \begin{Bmatrix} \varepsilon'_1 \\ \varepsilon'_2 \\ \varepsilon'_6 \\ \varepsilon'_4 \\ \varepsilon'_5 \end{Bmatrix} \qquad (2\text{-}16)$$

The Q_{ij}'s represent the ply stiffness expressed in material coordinates,

the σ'_i's are ply stresses, and the ε'_i's are the strains.

These relations assume plane stress conditions exist, therefore σ_3 is

negligible. Through the thickness strain (ε_3) is found through

constitutive relations as shown by Palazotto and Dennis (19). The Q_{ij}'s

in terms of engineering constants are:

$$Q_{11} = \frac{E_1}{1-\nu_{12}\nu_{21}} \quad Q_{12} = \frac{\nu_{12}E_2}{1-\nu_{12}\nu_{21}} = \frac{\nu_{21}E_1}{1-\nu_{12}\nu_{21}}$$

$$Q_{22} = \frac{E_2}{1-\nu_{12}\nu_{21}} \quad Q_{44} = G_{23} \; Q_{55} = G_{13} \; Q_{66} = G_{12} \qquad (2\text{-}17)$$

In these relations, E_i's are Young's moduli, G_{ij}'s are shear moduli,

and ν_{ij}'s are Poisson's ratios. All values are expressed in material

coordinates.

When constructing a laminate composite, the stiffness (Q_{ij}) of

each lamina must be converted to global coordinates. This

transformation is accomplished using the angle θ defined in figure 2-8

and the transformation equations found in Reddy (20). These equations

are presented in Appendix A and give the following constitutive

relations for the kth ply:

2-10

$$
\begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_6 \\ \sigma_4 \\ \sigma_5 \end{Bmatrix}_k =
\begin{bmatrix}
\overline{Q}_{11} & \overline{Q}_{12} & \overline{Q}_{16} & 0 & 0 \\
\overline{Q}_{12} & \overline{Q}_{22} & \overline{Q}_{26} & 0 & 0 \\
\overline{Q}_{16} & \overline{Q}_{26} & \overline{Q}_{66} & 0 & 0 \\
0 & 0 & 0 & \overline{Q}_{44} & 0 \\
0 & 0 & 0 & 0 & \overline{Q}_{55}
\end{bmatrix}_k
\begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_6 \\ \varepsilon_4 \\ \varepsilon_5 \end{Bmatrix}_k
\qquad (2\text{-}18)
$$

where $\overline{Q}_{ij}$'s represent the plane stress stiffnesses in global coordinates.

The next developmental step requires simplifying the 2-D Eqn(2-18) to 1-D. Because this work examines beams and arches which are narrow in the 1 direction, normal stresses in this direction σ_1 are negligible. Also, because the beam/arch is narrow and traction free at the sides, the in plane shear stress σ_6 is neglected. Stress and strain in the 5 direction (σ_5 and ε_5) are neglected because beam twisting is not considered. These assumptions reduce Eqn(2-18) to the following 1-D constitutive relations (3):

$$
\sigma_{4k} = \overline{Q}_{44k}\varepsilon_{4k}
$$
$$
\sigma_{2k} = \hat{Q}_{2k}\varepsilon_{2k}
$$

where $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (2\text{-}19)$

$$
\hat{Q}_{2k} = \left(\overline{Q}_{22} - \frac{\overline{Q}_{12}^2}{\overline{Q}_{11}} \right)_k
$$

Because twisted and torsional stiffness are not considered, Eqn(2-19) is limited to balanced, symmetric lay-ups. This limitation drives $\overline{Q}_{16}$ to zero for the entire laminate.

2.4 Strain Relations

Miller and Creaghan both employ the following relationship between Green strain and shell physical strains developed by Palazotto and Dennis (19).

$$\varepsilon_{ij} = \frac{\gamma_{ij}}{h_i h_j} \tag{2-20}$$

In Eqn(2-23), the γ_{ij}'s are Green strain tensor components and the h_i's are shell scale factors. The previously discussed thickness dependence of $\bar{u}_i$ drives the requirement for shell scale factors. Because only in plane normal and through the thickness shear strains are considered, only the two Green strain tensor terms related to these strains are required.

$$\gamma_{22} = h_2 u_{2,2} + \frac{h_2 u_3}{h_3} h_{2,3} + \frac{h_2 u_1}{h_1} h_{2,1}$$
$$+ \frac{1}{2}\left(u_{2,2} + \frac{u_3}{h_3} h_{2,3} + \frac{u_1}{h_1} h_{2,1}\right)^2 \tag{2-21}$$
$$+ \frac{1}{2}\left(u_{3,2} - \frac{u_2}{h_3} h_{2,3}\right)^2 + \frac{1}{2}\left(u_{1,2} - \frac{u_2}{h_1} h_{2,1}\right)^2$$

$$\gamma_{23} = \frac{1}{2}(h_3 u_{3,2} + h_2 u_{2,3} - u_2 h_{2,3} - u_3 h_{3,2})$$
$$+ \frac{1}{2}\left(u_{2,3} - \frac{u_3}{h_2} h_{2,3}\right)\left(u_{2,2} + \frac{u_3}{h_3} h_{2,3} + \frac{u_1}{h_1} h_{2,1}\right)$$
$$+ \frac{1}{2}\left(u_{3,2} - \frac{u_2}{h_3} h_{2,3}\right)\left(u_{3,3} + \frac{u_2}{h_2} h_{3,2} + \frac{u_1}{h_1} h_{3,1}\right) \tag{2-22}$$
$$+ \frac{1}{2}\left(u_{1,2} - \frac{u_2}{h_1} h_{2,1}\right)\left(u_{1,3} - \frac{u_2}{h_1} h_{3,1}\right)$$

Here, the $\bar{u}_i$'s come from Eqn(2-15).

The scale factors for a cylinder with curvature in only the 2 or s direction are:

$$h_1 = h_3 = 1$$
$$h_2 = 1 - \frac{z}{r}$$

(2-23)

Since beam width does not affect these scale factors, they also apply to a curved beam or arch. These factors match Eqn(2-9), which were developed graphically from figure 2-4.

Inserting Eqn(2-15), Eqn(2-21), and Eqn(2-23) into Eqn(2-20) yields the in plane normal strain ε_2 terms. All terms of Eqn(2-21) are retained because in-plane strain dominates thin beam bending problems. Representing the scale factors with truncated Taylor series simplifies expansion of Eqn(2-20). The resulting sixty expressions are shown in Palazotto and Dennis (19). For this work, only the following nine of the sixty scale factor approximations are required:

$$\frac{1}{h_2} \approx 1 + zc \qquad \frac{h_{2,3}}{h_2 h_3} \approx -c - c^2 z \qquad \frac{1}{h_2^2} \approx 1 + 2zc$$

$$\frac{h_{2,3}}{h_2^2 h_3} \approx -c - 2c^2 z \qquad \frac{h_{2,3}^2}{h_2 h_3} \approx c^2 + c^3 z \qquad \frac{h_{2,3}^2}{h_2^2} \approx c^2 + 2c^3 z$$

(2-24)

$$h_{2,3}^2 \approx c^2 \qquad \frac{h_{2,3}^2}{h_2} \approx c^2 + c^3 z \qquad \frac{h_{2,3}}{h_2^2} \approx -c - 2c^2 z$$

where $c = 1/r$. In plane normal strain is expressed in terms of the mid plane strain (ε_2^0) and functions independent of Z (χ_{2i}):

$$\varepsilon_2 = \varepsilon_2^0 + \sum_{p=1}^{7} z^p \chi_{2p}$$

(2-25)

The mid plane strain and χ_{2i}'s are defined as:

2-13

$$\varepsilon_2^0 = v_{,2} - wc + .5(v_{,2}^2 + w_{,2}^2 + v^2 c^2 + w^2 c^2) + vw_{,2}c - v_{,2}wc$$

$$\chi_{21} = \psi_{,2} - wc^2 + w_{,2}^2 c + w^2 c^3 - c^2(v_{,2}w - vw_{,2}) + v\psi c^2 + v_{,2}\psi_{,2}$$
$$- c(\psi_{,2}w - \psi w_{,2}) + \psi_{,2}\psi^2 + v_{,2}\psi_{,2}\psi^2 - wc\psi_{,2}\psi^2$$

$$\chi_{22} = \psi_{,2}c + .5(\psi_{,2}^2 + \psi^2 c^2) + v\psi c^3 - 2c^2(\psi_{,2}w - \psi w_{,2}) + \psi_{,2}v_{,2}c - 1.5(c^4 v^2 + c^2 v_{,2}^2)$$
$$- c^2 v_{,2} + 2c^3(v_{,2}w - vw_{,2}) + c(\psi_{,2}\psi^2 + \psi_{,2}\psi^2 v_{,2}) + \psi_{,2}\psi^2$$

$$\chi_{23} = k(w_{,22} + \psi_{,2}) + c\psi_{,2}^2 + \psi^2 c^3 + kv_{,2}(w_{,22} + \psi_{,2}) + vkc^2(w_{,2} + \psi) - wkc(w_{,22} + \psi_{,2})$$
$$+ w_{,2}kc(w_{,2} + \psi) + c^5 v^2 + c^3 v_{,2}^2 - 2c^2(c^2 v\psi + v_{,2}\psi_{,2}) \tag{2-26}$$

$$\chi_{24} = kc(w_{,22} + \psi_{,2}) + vkc^3(w_{,2} + \psi) + 2kc^2(-ww_{,22} - w_{,2}\psi + w_{,2}^2 + w\psi_{,2})$$
$$+ k\psi_{,2}(w_{,22} + \psi_{,2}) + \psi kc^2(w_{,2} + \psi) + v_{,2}kc(w_{,22} + \psi_{,2}) + k\psi_{,2}\psi^2(1 + w_{,22})$$

$$\chi_{25} = 2kc[\psi_{,2}(w_{,22} + \psi_{,2}) + \psi c^2(w_{,2} + \psi) - cv_{,2}(w_{,22} + \psi_{,2}) - c^3 v(w_{,2} + \psi) + \psi_{,2}\psi^2 w_{,22}]$$

$$\chi_{26} = \frac{k^2}{2}[w_{,22}^2 + 2w_{,22}\psi_{,2} + \psi_{,2}^2 + c^2(w_{,2}^2 + 2w_{,2}\psi + \psi^2)]$$

$$\chi_{27} = k^2 c[(w_{,22} + \psi_{,2})^2 + c^2(w_{,2} + \psi)^2]$$

In these equations, terms with combinations of c, ψ, $\psi_{,2}$, and $w_{,2}$ of order 5 and higher are neglected as higher order terms.

Because in-plane stress and strain dominate thin beam bending, through the thickness shear strain (ε_4) is developed using only the linear terms of Eqn(2-22). Combining these linear terms with Eqn(20) and Eqn(23) yields the following expression for through the thickness shear strain:

$$\varepsilon_4 = 2\varepsilon_{23} = \frac{1}{1 - \frac{z}{r}}\left[u_{3,2} + (1 - \frac{z}{r})u_{2,3} + \frac{u_2}{r}\right] \tag{2-27}$$

Small angle approximations are used in the global displacement equations because shear strain is linear, resulting in:

$$u_1 = 0$$
$$u_2(s, z) = v\left(1 - \frac{z}{r}\right) + z(\psi) + z^3 k(\psi + w_{,2}) \tag{2-28}$$
$$u_3(s) = w$$

Substituting Eqn(2-28) into Eqn(2-27) using $k = 4/(3h^2)$, one obtains:

$$\varepsilon_4 = \frac{1}{1 - \frac{z}{r}}\left[w_{,2} + \psi - \frac{4z^2}{h^2}(\psi + w_{,2}) + \frac{8z^3}{3h^2 r}(\psi + w_{,2})\right] \tag{2-29}$$

2-14

The scale factor term in front of Eqn(2-29) is approximated with a truncated binomial expansion:

$$\frac{1}{1-\frac{z}{r}} = 1+\frac{z}{r}+\frac{z^2}{r^2\left(1-\frac{z}{r}\right)}+...\approx 1 \tag{2-30}$$

The z^3 is also assumed negligible since $z \le$ h/2, leaving:

$$\varepsilon_4 = \varepsilon_4^0 + z^2\chi_{42}$$

where $\tag{2-31}$

$$\varepsilon_4^0 = w_{,2}+\psi \text{ and } \chi_{42} = \frac{-4}{h^2}(w_{,2}+\psi)$$

as the expressions for through the thickness shear strain.

2.5 Beam Potential Energy

The potential energy Π_p is defined as the sum of the internal strain energy (U) and the external work (V):

$$\Pi_p = U + V \tag{2-32}$$

The internal strain energy is defined as:

$$U = \int_{vol} W^* dV \tag{2-33}$$

where W^* is the strain energy density function:

$$W^* = \frac{1}{2}a_{ijkl}\varepsilon_{ij}\varepsilon_{kl} \tag{2-34}$$

The a_{ijkl} term is an elasticity tensor.

In this case, the internal strain energy is made up of two components, the in-plane strain (U_1) and the through the thickness shear strain (U_2):

$$U_1 = \frac{1}{2}b\int_l \int_{-\frac{h}{2}}^{\frac{h}{2}} \hat{Q}_{2k}\varepsilon_2^2 dzds \tag{2-35}$$

2-15

$$U_2 = \frac{1}{2} b \int_l \int_{-\frac{h}{2}}^{\frac{h}{2}} \overline{Q}_{44k} \varepsilon_4^2 \, dz \, ds \qquad (2\text{-}36)$$

Using Eqn(2-25):

$$\varepsilon_2^2 = (\varepsilon_2^0)^2 + 2 \sum_{p=1}^{7} z^p \chi_{2p} \varepsilon_2^0 + \sum_{p=1}^{7} (z^p \chi_{2p})^2 \qquad (2\text{-}37)$$

Now U_1 is divided into three parts, each representing a strain component from Eqn(2-37):

$$U_1 = \frac{1}{2} b \int_l (\mu_1 + \mu_2 + \mu_3) \qquad (2\text{-}38)$$

where:

$$\mu_1 = \int_{-\frac{k}{2}}^{\frac{k}{2}} \hat{Q}_{2k} (\varepsilon_2^0)^2 \, dz = A(\varepsilon_2^0)^2$$

$$\mu_2 = \int_{-\frac{h}{2}}^{\frac{h}{2}} 2\hat{Q}_{2k} (\varepsilon_2^0) \sum_{p=1}^{7} \chi_{2p} z^p \, dz = 2\varepsilon_2^0 B\chi_{21} + D\chi_{22} + E\chi_{23} + F\chi_{24} + G\chi_{25} + H\chi_{26} + I\chi_{27})$$

$$\mu_3 = \int_{-\frac{h}{2}}^{\frac{h}{2}} \hat{Q}_{2k} \left(\sum_{p=1}^{7} \chi_{2p} z^p \right)^2 \, dz$$

$$\begin{aligned}
= D\chi_{21}^2 &+ 2E\chi_{21}\chi_{22} + F(2\chi_{21}\chi_{23} + \chi_{22}^2) + 2G(\chi_{21}\chi_{24} + \chi_{22}\chi_{23}) + 2H(\chi_{21}\chi_{25} + \chi_{22}\chi_{24}) \quad (2\text{-}39) \\
&+ 2I(\chi_{21}\chi_{26} + \chi_{22}\chi_{25} + \chi_{23}\chi_{24}) + J(2\chi_{21}\chi_{27} + 2\chi_{22}\chi_{26} + 2\chi_{23}\chi_{25} + \chi_{24}^2) \\
&+ 2K(\chi_{22}\chi_{27} + \chi_{23}\chi_{26} + \chi_{24}\chi_{25}) + L(2\chi_{23}\chi_{27} + 2\chi_{24}\chi_{26} + \chi_{25}^2) + 2P(\chi_{24}\chi_{27} + \chi_{25}\chi_{26}) \\
&+ R(2\chi_{25}\chi_{27} + \chi_{26}^2) + 2S\chi_{26}\chi_{27} + T\chi_{27}^2
\end{aligned}$$

The μ_i's of Eqn(2-39) have been integrated through the thickness such that:

$$(A, B, D, E, F, G, H, I, J, K, L, P, R, S, T) =$$
$$\int_{-\frac{h}{2}}^{\frac{h}{2}} \hat{Q}_{2k} \left(1, z, z^2, z^3, z^4, z^5, z^6, z^7, z^8, z^9, z^{10}, z^{11}, z^{12}, z^{13}, z^{14} \right) dz \qquad (2\text{-}40)$$

All other terms of Eqn(2-39) are independent of thickness. If composite materials are used, the balanced symmetric lay up drives the integration of any odd powers of z to zero.

Strain energy due to through the thickness shear is similarly derived. In this case:

$$U_2 = \frac{1}{2}b\int_l \mu_s ds$$

where: (2-41)

$$\mu_s = \int_{-\frac{h}{2}}^{\frac{h}{2}} \overline{Q}_{44k}[(\varepsilon_4^0)^2 + 2\varepsilon_4^0\chi_{42}z^2 + (\chi_{42})^2 z^4]dz$$

$$= AS(\varepsilon_4^0)^2 + 2DS\varepsilon_4^0\chi_{42} + FS(\chi_{42})^2$$

Again, the elasticity terms (AS,DS,FS) are found by integrating through the thickness:

$$[AS, DS, FS] = \int_{-\frac{h}{2}}^{\frac{h}{2}} \overline{Q}_{44k}[1, z^2, z^4]dz$$ (2-42)

The strain expressions of Eqn(2-25) and Eqn(2-31) are divided into linear and nonlinear parts to facilitate placing them into the preceding strain energy formulations. This is done by first defining the displacement gradient vector **d** such that:

$$\mathbf{d}^T = \left\{ v \quad v_{,2} \quad w \quad w_{,2} \quad w_{,22} \quad \psi \quad \psi_{,2} \right\}$$ (2-43)

Now the in plane strain can be expressed as:

$$\varepsilon_2^0 = \mathbf{L}_0^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T\mathbf{H}_0\mathbf{d}$$

$$\chi_{2p} = \mathbf{L}_p^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T\mathbf{H}_p\mathbf{d}$$ (2-44)

where p = 1,2,3,4,5,6,7. The $\mathbf{L}_i$'s are column vectors and the $\mathbf{H}_i$'s are symmetric matrices. Combining the $\mathbf{L}_i$'s and $\mathbf{H}_i$'s with the displacement gradient vector creates the linear and nonlinear strain terms, respectively. This is demonstrated for the mid plane strain term (ε_2^0):

2-17

$$\varepsilon_2^0 = \begin{Bmatrix} 0 & 1 & -c & 0 & 0 & 0 & 0 \end{Bmatrix} \begin{Bmatrix} v \\ v_{,2} \\ w \\ w_{,2} \\ w_{,22} \\ \psi \\ \psi_{,2} \end{Bmatrix}$$

$$+ \frac{1}{2} \begin{Bmatrix} v & v_{,2} & w & w_{,2} & w_{,22} & \psi & \psi_{,2} \end{Bmatrix} \begin{bmatrix} c^2 & 0 & 0 & c & 0 & 0 & 0 \\ 0 & 1 & -c & 0 & 0 & 0 & 0 \\ 0 & -c & c^2 & 0 & 0 & 0 & 0 \\ c & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} v \\ v_{,2} \\ w \\ w_{,2} \\ w_{,22} \\ \psi \\ \psi_{,2} \end{Bmatrix} \qquad (2\text{-}45)$$

The through the thickness shear strain (ε_4) expressions are less complicated due to the previous assumptions about linearity and small angle approximations:

$$\varepsilon_4^o = S_0^T \mathbf{d}$$
$$\chi_{42} = S_2^T \mathbf{d} \qquad (2\text{-}46)$$

Appendix A contains the complete **H**, **L**, and **S** matrices.

Miller's derivation of the $\mathbf{H}_i$ matrices contains non linear terms of ψ whereas Creaghan's $\mathbf{H}_i$'s contained only constants. Miller treats the $\mathbf{H}_i$'s as constants to maintain similarity with Creaghan and Palazotto and Dennis. This linearization is accomplished by substituting the previous incremental value of ψ into the current $\mathbf{H}_i$'s (3; 13; 19). Although this assumption simplifies the impending variational calculus, it limits the accuracy of Miller's theory because the behavior of the bending angle is not properly modeled and leads to inexact equilibrium equations. This may be a major reason the theory is limited to moderate rotations (46 degrees). Subsequent research should investigate this assumption.

The total beam strain energy can now be expressed as:

$$U = \frac{1}{2} b \int_l \mathbf{d}^T \left[\hat{\mathbf{K}} + \frac{\hat{\mathbf{N}}_1(\mathbf{d})}{3} + \frac{\hat{\mathbf{N}}_2(\mathbf{d})}{6} \right] \mathbf{d}\, ds \qquad (2-47)$$

where $\hat{\mathbf{K}}$ is a matrix of constants, $\hat{\mathbf{N}}_1$ is a matrix of the linear functions of $\mathbf{d}$, and $\hat{\mathbf{N}}_2$ is a matrix containing quadratic terms from $\mathbf{d}$. This expression is formed by substituting Eqns(2-44) and (2-46) into Eqns(2-38), (2-39), (2-41), and (2-42) as outlined by Palazotto and Dennis (18). This form of the equation simplifies finite element formulation. As previously stated, the higher order terms of ψ are treated as constants in $\hat{\mathbf{N}}_1$ and $\hat{\mathbf{N}}_2$. The $\hat{\mathbf{K}}$, $\hat{\mathbf{N}}_1$, and $\hat{\mathbf{N}}_2$ matrices are listed in Appendix A.

2.6 Finite Element Formulation

The beam finite element used for the current formulation is shown in figure 2-9. The element has three nodes with degrees of freedom (DOF) v, w, $w_{,2}$, ψ, and $\psi_{,2}$ at the end nodes and only v at the middle node. This element is similar to Miller's except for the addition of the $\psi_{,2}$ degree of freedom at the end nodes.

The additional DOF facilitates Hermitian shape functions for the bending angle (ψ). This element maintains C^1 continuity for w and ψ and their derivatives and C^0 continuity for v. Previous efforts only maintained C^0 continuity for ψ.

Adding two DOF also mandates additional boundary conditions. This element experiences clamped, hinged, free, and symmetric boundary conditions and $\psi_{,2}$ must be defined for each. Because a physical
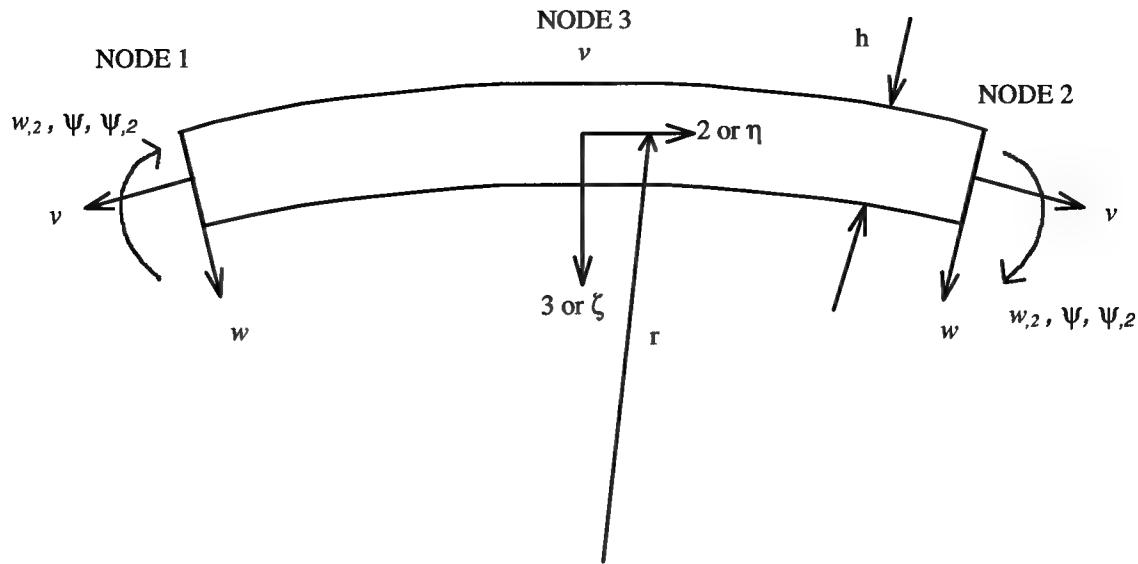
Figure 2-9 Beam Finite Element

interpretation of $\psi_{,2}$ is impossible, the values for $\psi_{,2}$ under these conditions are derived mathematically from:

$$\psi_{,2} \approx \frac{\partial^2 w}{\partial s^2} \approx \frac{M}{EI} \qquad (2-48)$$

Based on this assumption, $\psi_{,2} = 0$ at any boundary condition incapable of supporting a bending moment. Table 2-2 lists the resulting boundary conditions used for the DOF at nodes 1 and 2. For the current work, v at node 3 is never specified.

Table 2-2 Specified Boundary Conditions

	v	w	$w_{,2}$	ψ	$\psi_{,2}$
clamped	0	0	0	0	free
hinged	0	0	free	free	0
free	free	free	free	free	0
point of symmetry	0	0	0	0	free

The values of the DOF $\mathbf{q}(\eta)$ at any point can be expressed in natural coordinates in terms of the element's DOF using:

$$
\mathbf{q}(\eta)=\left\{\begin{array}{c} v \\ \psi \\ \psi_{,2} \\ w \\ w_{,2} \end{array}\right\}=
\left[\begin{array}{ccccccccccc}
Q_1 & 0 & 0 & 0 & 0 & Q_3 & Q_2 & 0 & 0 & 0 & 0 \\
0 & H_{11} & H_{12} & 0 & 0 & 0 & 0 & H_{21} & H_{22} & 0 & 0 \\
0 & H_{11,\eta} & H_{12,\eta} & 0 & 0 & 0 & 0 & H_{21,\eta} & H_{22,\eta} & 0 & 0 \\
0 & 0 & 0 & H_{11} & H_{12} & 0 & 0 & 0 & 0 & H_{21} & H_{22} \\
0 & 0 & 0 & H_{11,\eta} & H_{12,\eta} & 0 & 0 & 0 & 0 & H_{21,\eta} & H_{22,\eta}
\end{array}\right]
\left\{\begin{array}{c}
v^{(1)} \\ \psi^{(1)} \\ \psi_{,2}^{(1)} \\ w^{(1)} \\ w_{,2}^{(1)} \\ v^{(3)} \\ v^{(2)} \\ \psi^{(2)} \\ \psi_{,2}^{(2)} \\ w^{(2)} \\ w_{,2}^{(2)}
\end{array}\right\}
$$

$$(2\text{-}49)$$

where the shape functions are:

$$
Q_1 = \frac{1}{2}(\eta^2 - \eta) \quad Q_2 = \frac{1}{2}(\eta^2 + \eta) \quad Q_3 = (1-\eta^2)
$$

$$
H_{11} = \frac{1}{4}(2-3\eta+\eta^3) \quad H_{12} = \frac{a}{4}(1-\eta-\eta^2+\eta^3)
$$

$$
H_{21} = \frac{1}{4}(2+3\eta-\eta^3) \quad H_{22} = \frac{a}{4}(-1-\eta+\eta^2+\eta^3)
$$

$$(2\text{-}50)$$

The displacement gradient vector $\mathbf{d}$ of Eqn(2-43) must be related to the DOF before using the beam energy equation, Eqn(2-47). The conversion is done in natural coordinates using the following equation:

$$
\mathbf{d}(\eta)=\mathbf{D}\mathbf{d}=
\left[\begin{array}{ccccccccccc}
Q_1 & 0 & 0 & 0 & 0 & Q_3 & Q_2 & 0 & 0 & 0 & 0 \\
Q_{1,\eta} & 0 & 0 & 0 & 0 & Q_{3,\eta} & Q_{2,\eta} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & H_{11} & H_{12} & 0 & 0 & 0 & 0 & H_{211} & H_{22} \\
0 & 0 & 0 & H_{11,\eta} & H_{12,\eta} & 0 & 0 & 0 & 0 & H_{21,\eta} & H_{22,\eta} \\
0 & 0 & 0 & H_{11,\eta\eta} & H_{12,\eta\eta} & 0 & 0 & 0 & 0 & H_{21,\eta\eta} & H_{22,\eta\eta} \\
0 & H_{11} & H_{12} & 0 & 0 & 0 & 0 & H_{21} & H_{22} & 0 & 0 \\
0 & H_{11,\eta} & H_{12,\eta} & 0 & 0 & 0 & 0 & H_{21,\eta} & H_{22,\eta} & 0 & 0
\end{array}\right]
\left\{\begin{array}{c}
v^{(1)} \\ \psi^{(1)} \\ \psi_{,2}^{(1)} \\ w^{(1)} \\ w_{,2}^{(1)} \\ v^{(3)} \\ v^{(2)} \\ \psi^{(2)} \\ \psi_{,2}^{(2)} \\ w^{(2)} \\ w_{,2}^{(2)}
\end{array}\right\}
$$

$$(2\text{-}51)$$

where **D** is the shape function matrix used to relate the **d** to the DOF. By multiplying by the inverse of the Jacobian matrix (**J**), Eqn(2-51) is converted to global coordinates.

$$\mathbf{d}(s) = \mathbf{J}^{-1}\mathbf{d}(h) = \mathbf{J}^{-1}\mathbf{D}\mathbf{q} = \mathbf{T}\mathbf{q} \tag{2-52}$$

where $\mathbf{J}^{-1}$ is defined as:

$$\mathbf{J}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{1}{a} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \dfrac{1}{a} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{1}{a^2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \dfrac{1}{a} \end{bmatrix} \tag{2-53}$$

where a is half the element length.

Now the beam strain energy from Eqn(2-47) based on the DOF is:

$$U = \frac{1}{2}b\int_{l} \mathbf{q}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\left[\hat{\mathbf{K}} + \frac{\hat{\mathbf{N}}_2}{3} + \frac{\hat{\mathbf{N}}_2}{6}\right]\mathbf{T}\mathbf{q}\,ds \tag{2-54}$$

Integrating along the beam length yields new terms used for a tangent stiffness matrix:

$$\mathbf{K} = b\int_{l}\mathbf{T}^{\mathrm{T}}\hat{\mathbf{K}}\mathbf{T}ds \cdot \mathbf{N}_1 = b\int_{l}\mathbf{T}^{\mathrm{T}}\hat{\mathbf{N}}_1\mathbf{T}ds \cdot \mathbf{N}_2 = b\int_{l}\mathbf{T}^{\mathrm{T}}\hat{\mathbf{N}}_2\mathbf{T}ds \tag{2-55}$$

Substituting Eqn(2-55) into Eqn(2-54), one obtains this element's potential energy equation:

$$\Pi_p = \frac{1}{2}\mathbf{q}^{\mathrm{T}}\left[\mathbf{K} + \frac{\mathbf{N}_1(\mathbf{q})}{3} + \frac{\mathbf{N}_2(\mathbf{q})}{6}\right]\mathbf{q} - \mathbf{q}^{\mathrm{T}}\mathbf{R} \tag{2-56}$$

where **R** is a vector representation of external forces.

Defining the beam equilibrium equation requires taking the first variation of Eqn(2-56) and setting it equal to zero:

2-22

$$\delta\Pi_p = \delta\mathbf{q}^T\left[\left[\mathbf{K} + \frac{\mathbf{N}_1(\mathbf{q})}{2} + \frac{\mathbf{N}_2(\mathbf{q})}{3}\right]\mathbf{q} - \mathbf{R}\right] = \delta\mathbf{q}^T[\mathbf{F}(\mathbf{q})] = 0 \qquad (2\text{-}57)$$

where F is the resultant equilibrium equations. The first variation is
accomplished with respect to the vertical displacement (*w*) because the
bending angle ψ is taken to be a linearized function and the membrane
displacement (v) is a linear function. This is where the theory goes
wrong. The first variation should be accomplished with respect to the
bending angle which includes a third order approximation as well as the
vertical displacement. The linearization of ψ limits the exactness of
the equilibrium equations as discussed in section 2.5.

Because $\delta\mathbf{q}$ is arbitrary and independent, $\mathbf{F}(\mathbf{q}) = 0$ becomes the set
of algebraic nonlinear equilibrium equations. To solve these equations,
Eqn(2-57) is expanded in a truncated Taylor series
such that:

$$\mathbf{F}(\mathbf{q}+\delta\mathbf{q}) \approx \mathbf{F}(\mathbf{q}) + \frac{\partial \mathbf{F}}{\partial \mathbf{q}}\delta\mathbf{q} = 0$$

or: $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ (2-58)

$$\frac{\partial \mathbf{F}}{\partial \mathbf{q}}\delta\mathbf{q} = -\mathbf{F}(\mathbf{q})$$

Substituting the expression for F and its derivatives into Eqn(2-58)
yields:

$$\mathbf{K}_T\delta\mathbf{q} = -\left[\mathbf{K} + \frac{\mathbf{N}_2}{2} + \frac{\mathbf{N}_2}{3}\right]\mathbf{q} + \mathbf{R}$$

where: $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ (2-59)

$$\mathbf{K}_T = [\mathbf{K} + \mathbf{N}_1 + \mathbf{N}_2]$$

Eqn(2-59) is the final equation used to find the equilibrium values for
the 11 DOF. In any case, the nonlinear stiffness matrices vary with
load and displacement throughout the structure.

2.7 Numerical Solution Methods

Modifying Eqn(2-59) to include all elements of a finite element model, one obtains:

$$\sum_{j=1}^{n}\left[b\int_l \mathbf{T}^{\mathrm{T}}\left(\hat{\mathbf{K}}+\hat{\mathbf{N}}_1+\hat{\mathbf{N}}_2\right)\mathbf{T}ds\right]_j \delta\mathbf{q} = -\sum_{j=1}^{n}\left[b\int_l \mathbf{T}^{\mathrm{T}}\left(\hat{\mathbf{K}}+\frac{\hat{\mathbf{N}}_1}{2}+\frac{\hat{\mathbf{N}}_2}{3}\right)\mathbf{T}ds\right]_j \mathbf{q}+\mathbf{R} \qquad (2\text{-}60)$$

where n is the number of elements in the model and j is the integration over the length of any particular element. The global load vector $\mathbf{R}$ contains as many rows as DOF. The $\mathbf{q}$ and $\delta\mathbf{q}$ vectors are global displacement arrays constructed from the elemental displacement vectors.

Eqn(2-60) is integrated using Gauss quadrature. Although exact integration would require 7th order Gauss quadrature, 5th order quadrature is used with negligible difference (8; 13).

Miller employed the Newton-Raphson solution methods which are retained for the current work. Either the displacement control method developed by Dennis and Palazotto or a modified Riks-Wempner technique is used to generate equilibrium paths (13; 19; 27). Both techniques are briefly reviewed here.

The displacement control method iterates load values based on incremental displacements until equilibrium is achieved. The initial displacement as well as each displacment increment are prescribed. This method transverses horizontal limit points similar to point A in figure 2-10; however, vertical limits such as point B cannot be evaluated because no unique solution exists at that displacement. Figure 2-11 graphically depicts the displacement control method. Starting at an equilibrium point, the displacement is incremented and a load determined using:

$$\left[\mathbf{K}+\mathbf{N}_1\left(\mathbf{q}_{n-1}\right)+\mathbf{N}_2\left(\mathbf{q}_{n-1}^2\right)\right]\delta\mathbf{q}_{n-1} = -\left[\mathbf{K}+\frac{\mathbf{N}_1\left(\mathbf{q}_{n-1}\right)}{2}+\frac{\mathbf{N}_2\left(\mathbf{q}_{n-1}^2\right)}{3}\right]\mathbf{q}_{n-1}+\mathbf{R}_1 \qquad (2\text{-}61)$$
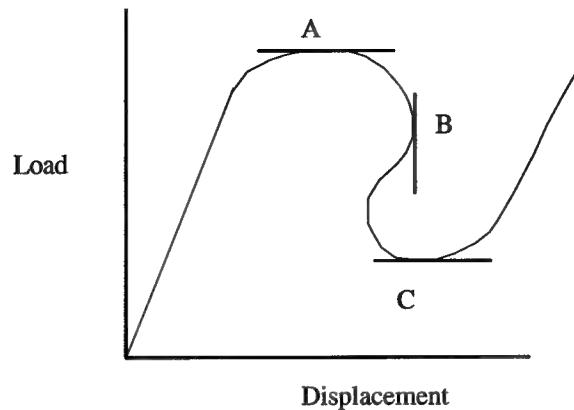
Figure 2-10   Typical Load vs. Displacement Curve
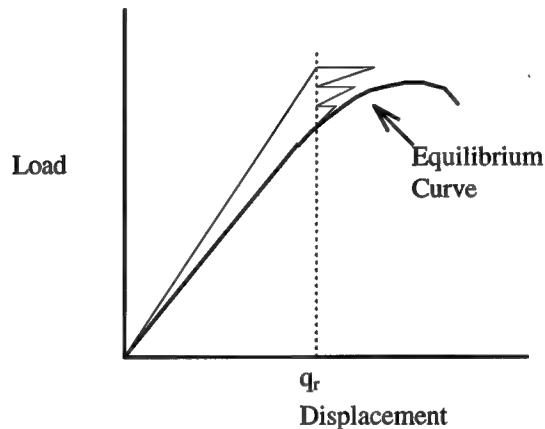
Figure 2-11 Displacement Control

Eqn(2-61) comes from Palazotto and Dennis (19).   Iteration of Eqn(2-61) is continued until convergence, for a given displacement increment, satisfies equilibrium

The Tsai and Palazotto modified Riks-Wempner technique transverses limit points the displacement control method cannot (27).   This technique, which is referred to as the Riks method, modifies the equilibrium equation such that:

$$F(q, \lambda) = \left(K + \frac{N_1}{2} + \frac{N_2}{3}\right)q - \lambda R \qquad (2-62)$$

where λ is a user specified loading parameter. Eqn(2-62) is expanded for a solution similar to Eqn(2-56), resulting in:

$$\mathbf{K}_T \delta \mathbf{q}_i = \delta \Delta_i \mathbf{R} - \mathbf{F}(\mathbf{q}_i, \lambda_i) \tag{2-63}$$

A search radius, Δl is defined as the additional constraint required for Eqn(2-63) to be a solvable system. This value is defined as:

$$\Delta l^2 = \Delta \mathbf{q}_{i+1}^T \Delta \mathbf{q}_{i+1} + \Delta \lambda_{i+1}^2 \mathbf{R}^T \mathbf{R} \tag{2-64}$$

The search radius establishes the distance of the new equilibrium point from the current equilibrium point. Assuming Δl is small enough to accurately capture the equilibrium path, it may be approximated using:

$$\Delta l^2 = \Delta \mathbf{q}_{i+1}^T \Delta \mathbf{q}_{i+1} \tag{2-65}$$

This definition causes an unsymmtrical global stiffness matrix. Reducing $\delta \mathbf{q}_i$ to two terms restores the required symmetry:

$$\delta \mathbf{q}_i = \delta \mathbf{q}_{i1} + \delta \lambda_i \delta \mathbf{q}_{i2}$$

where $\hspace{10cm}$ (2-66)

$$\delta \mathbf{q}_{i1} = -\mathbf{K}_T^{-1} \mathbf{F}(\mathbf{q}_i, \lambda_i), \quad \delta \mathbf{q}_{i2} = -\mathbf{K}_T^{-1} \mathbf{R}$$

Each incremental displacement and load increment values are based on the results of the previous increment and a small change δ:

$$\Delta \mathbf{q}_{i+1} = \Delta \mathbf{q}_i + \delta \mathbf{q}_{i+1}$$
$$\Delta \lambda_{i+1} = \Delta \lambda_i + \delta \lambda_i \tag{2-67}$$

Substituting Eqn(2-67) into Eqn(2-65) yields:

$$a \delta \lambda_i^2 + b \delta \lambda_i + c = 0$$

where $\hspace{10cm}$ (2-68)

$$a = \delta \mathbf{q}_{i2}^T \delta \mathbf{q}_{i2}, \quad b = 2 \delta \mathbf{q}_{i2}^T (\Delta \mathbf{q}_i + \delta \mathbf{q}_i)$$
$$c = (\Delta \mathbf{q}_i + \delta \mathbf{q}_i)^T (\Delta \mathbf{q}_i + \delta \mathbf{q}_i) - \Delta l^2$$

The preceding system of equations converges when $\delta \mathbf{q}_i$ becomes smaller

2-26

than a user defined value. After convergence at load step m, the total

displacement $\mathbf{q}_m$, the total load parameter λ_m, and the incremental load

step Δl_m are calculated using:

$$\mathbf{q}_m = \mathbf{q}_{m-1} + \Delta\mathbf{q}_m$$
$$\lambda_m = \lambda_{m-1} + \Delta\lambda_m \qquad\qquad (2\text{-}69)$$
$$\Delta l_m = \Delta l_{m-1} \frac{N_s}{N_{m-1}}$$

where N_s is a user defined number of iteration prediction and N_{m-1} the

number of iterations required for convergence of step $m\text{-}1$. Once the

search radius for a step is defined, the initial load parameter is found

using:

$$\Delta\lambda_1 = \pm \frac{\Delta l_m}{\left(\delta\mathbf{q}_{i2}^{\mathrm{T}}\delta\mathbf{q}_{i2}\right)^{1/2}} \qquad\qquad (2\text{-}70)$$

2.8 Step by Step Riks Algorithm

This Riks method example is presented by Creaghan and Miller and

is repeated here for clarity (3; 13). Figure 2-12 graphically depicts

the process. For each Riks solution, the estimated number of iterations

(N_s), total number of steps (m), the incremental displacement limit

$(\mathbf{q}_i)$, and the initial load parameter (λ_0) are user prescribed parameters.

1. First increment, first iteration: compute only the constant

    stiffness matrix, $\mathbf{K}$.

2. First iteration, subsequent increments: compute $\delta\mathbf{q}_{i2} = -\mathbf{K}_T^{-1}\mathbf{R}$.

    After the first increment, $\mathbf{K}_T$ is composed from $\mathbf{q}_{m-1}$ nonlinear

    displacement terms.

3. Each iteration, each increment: compute $\Delta\mathbf{q}_i = \Delta\lambda_i\delta\mathbf{q}_{i2}$. For the first

    increment, $\Delta\lambda_i = \lambda_0$, which is a program input. Current examples
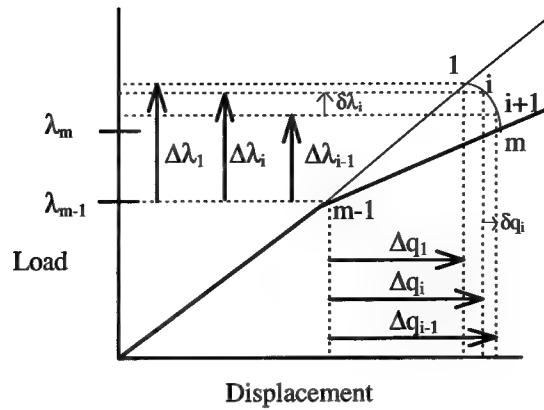
Figure 2-12 Riks Method

used $\lambda_0 = 0.1$ as did Creaghan and Miller with acceptable results (3; 13). For subsequent increments, $\Delta\lambda_i = \lambda_1$.

4. Each iteration, each increment: compute in order:

$$\delta\mathbf{q}_{i1} = -\mathbf{K}_T^{-1}\,\mathbf{F}(\mathbf{q}_i,\lambda_i)$$
$$\delta\mathbf{q}_i = \delta\mathbf{q}_{i1} + \delta\lambda_i\,\delta\mathbf{q}_{i2} \qquad\qquad (2\text{-}71)$$
$$\Delta\mathbf{q}_{i+1} = \Delta\mathbf{q}_i + \delta\mathbf{q}_{i+1}$$

5. Compute $\Delta l = \left(\Delta\mathbf{q}_{i+1}^{T}\Delta\mathbf{q}_{i+1}\right)$.

6. Update $\mathbf{K}^T$ with $\mathbf{q}_i = \mathbf{q}_{m-1} + \Delta\mathbf{q}_i$.

7. Solve Eqn(2-68) for $\pm\delta\lambda_i$. For complex roots, return to step 2 and arbitrarily adjust Δl_m, which changes $\Delta\lambda_1$.

8. Choose $\pm\delta\lambda_i$ based on which value yields a positive θ in:

$$\delta\mathbf{q}_i = \delta\mathbf{q}_1 \pm \delta\lambda_i\,\delta\mathbf{q}_{i2}$$
$$\theta = \left(\Delta\mathbf{q}_i + \delta\mathbf{q}_i\right)\Delta\mathbf{q}_i \qquad\qquad (2\text{-}72)$$

If both are positive, then $\delta\lambda_i = -c/b$.

9. Update the displacement and loading parameters using:

$$\Delta\mathbf{q}_{i+1} = \Delta\mathbf{q}_i + \delta\mathbf{q}_{i+1}$$
$$\Delta\lambda_{i+1} = \Delta\lambda_i + \delta\lambda_i \qquad\qquad (2\text{-}73)$$

2-28

10. Check for convergence. If no convergence, return to step 2. upon

convergence, update the displacement, load parameter, and search

radius for the next increment:

$$\mathbf{q}_m = \mathbf{q}_{m-1} + \Delta\mathbf{q}_m$$
$$\lambda_m = \lambda_{m-1} + \Delta\lambda_m \qquad\qquad (2\text{-}74)$$
$$\Delta l_m = \Delta l_{m-1}\frac{N_s}{N_{m-1}}$$

where N_s is a user defined number of iteration prediction and N_{m-1}

the number of iterations required for convergence of step $m\text{-}1$.

The current work uses $N_s = 2.5$.

11. Compute the loading parameter for the first iteration of the next

increment:

$$\Delta\lambda_1 = \pm\frac{\Delta l_m}{\left(\delta\mathbf{q}_{i2}^T\delta\mathbf{q}_{i2}\right)^{\frac{1}{2}}} \qquad\qquad (2\text{-}75)$$

12. Return to step 2 to begin the next increment.

2.9 Arch Geometry Considerations

As previously discussed, geometry plays a significant role in the

arch behavior. Figure 2-13 presents a typical arch geometry. Miller
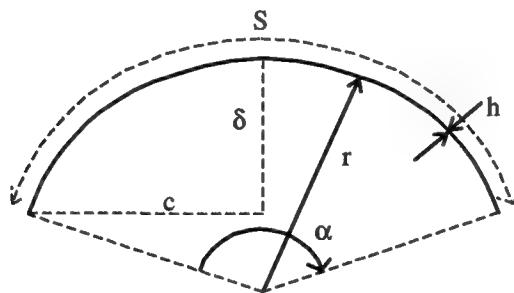
Figure 2-13 Arch Geometry Definitions

considers an arch thick when (12):

$$\frac{c}{h} < 25 \qquad (2-76)$$

He also defines a deep arch using:

$$\lambda \geq 8$$

where $\qquad (2-77)$

$$\lambda = \left(12\left(1-v^2\right)\right)^{\frac{1}{4}}\left(\frac{r}{h}\right)^{\frac{1}{2}}\frac{\alpha}{2}$$

These classification will be used when comparing the current work with Miller's.

III. Results and Discussion

3.1 Overview

This chapter is divided into two sections. In the first section, the current work is compared to Miller's work to assess the effects of the Hermitian shape functions. The second section presents the results of extensive arch analysis using the current program.

3.2 Comparison

3.2.1 Clamped Isotropic Shallow Thin Arch.
The first problem considered was originally presented by Belytschko and Glaum(1) and is

$\alpha = 17.2°$
$r = 3.381 m$
$h = 4.763 \times 10^{-3} m$
$\delta/c = 0.075$
$E = 68.946 \, GPa$
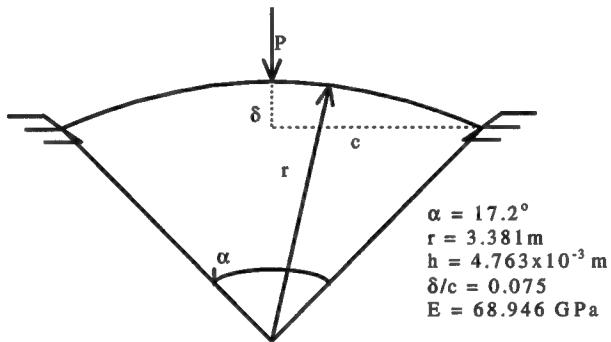
Figure 3-1 Clamped Isotropic Shallow Thin Arch

presented here in figure 3-1. Miller and Creaghan also analyze this structure(3; 13). Because the current work is an evolution of Miller's, comparisons are only made with his results.

This first example combines large displacements and geometric nonlinearity with small rotations. As such, it is used to validate the current work and demonstrate that the updated shape functions do not corrupt the program's capabilities. The structure is symmetrically
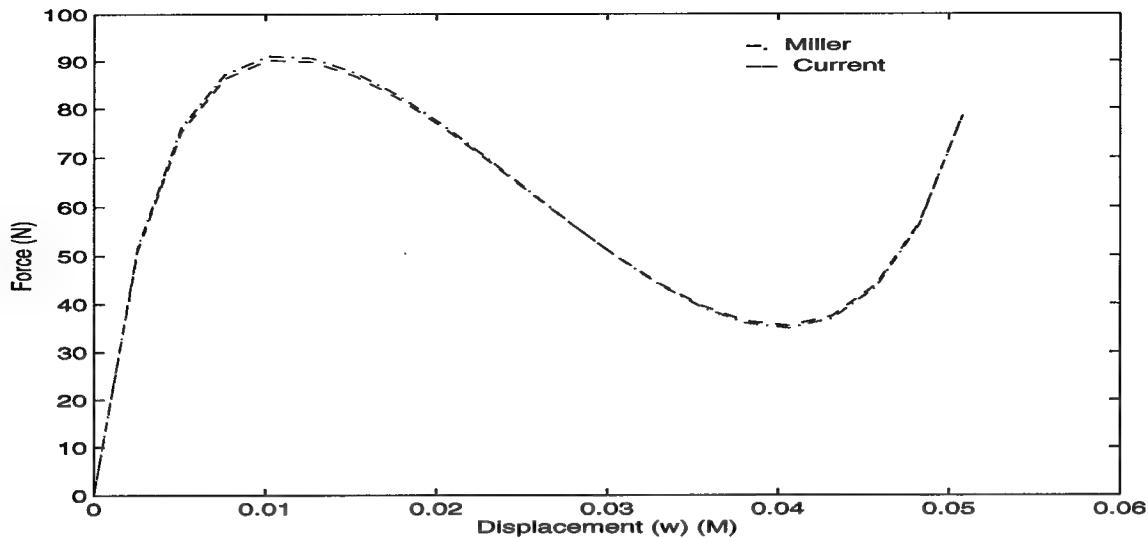
Figure 3-2 Comparison for Clamped Shallow Thin Arch

modeled using 17 elements. Miller's model contained 81 active degrees of freedom (DOF) while the current work used 99. Both formulations use displacement control with a convergence tolerance of 0.1% and maximum of 200 iterations per displacement increment. Twenty increments are used with an initial displacement of 2.54mm. Convergence is based on:

$$\frac{\sqrt{\sum_i \left(\mathbf{q}_r^i\right)^2} - \sqrt{\sum_i \left(\mathbf{q}_{r-1}^i\right)^2}}{\sqrt{\sum_i \left(\mathbf{q}_1^i\right)^2}} \times 100\% \leq TOL \tag{3-1}$$

where $\mathbf{q}_r^i$, $\mathbf{q}_{r-1}^i$, and $\mathbf{q}_1^i$ are elements of the global displacement vector, for the rth, $(r - 1)^{th}$, and the first load increment, $i = 1$ to n degrees of freedom, and TOL is the user defined tolerance (19).

Figure 3-2 compares the vertical displacement (w) at node 8 (the point of symmetry where P is applied) for each solution method. The maximum difference is less than 1.0% at 10.16mm of displacement. This figure is attained using:

$$difference = \left| \frac{Miller's - Current}{Miller's} \right| \times 100 \tag{3-2}$$

This confirms correct introduction of Hermitian shape functions into the finite element formulation.

Examining the derivative of the bending angle ($\psi_{,2}$) demonstrates the effect of the higher order shape function. Figure 3-3 compares ($\psi_{,2}$) for the current work with Miller's results. The linear shape
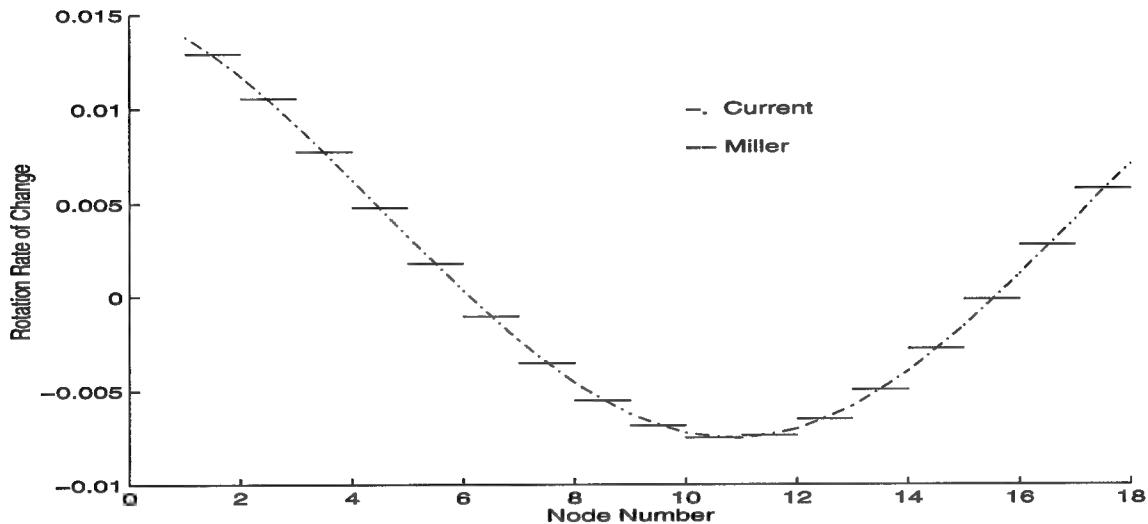
Figure 3-3 Clamped Shallow Thin Arch Rotation Rate of Change Comparison

function of the previous effort create a curve with step discontinuities at each node. The current work provides a continuous curve for the bending angle rate of change expression. Thus, the higher order shape function is indeed properly incorporated and does not degrade the capabilities of the previous effort.

To examine the efficiency of the two programs, the number of elements used to model the structure is incrementally reduced until the solutions deviated from the "approved" solution presented in figure 3-2 by more than 1.0%. Both formulations were run using the same setup as the initial solution with the exception of the reduced number of elements. Using 12 elements and 56 active DOF, Miller's program

deviated 0.848%. Using this model, his program required 23.1 seconds to solve using a Sun Sparc20 workstation. The current work required only 7 elements and 39 active DOF to achieve similar results, using the same convergence parameters. This setup required only 17.3 seconds to run on the same workstation, a 25.1% reduction.

3.2.2 Cantilever Isotropic Thin Beam. Figure 3-4 presents the second problem examined. This structure experiences large deflections
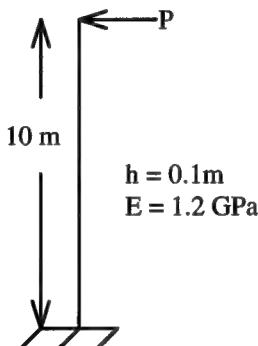
Figure 3-4 Cantilever Isotropic Beam

and moderate rotations. Hsiao and Hou initially examine this beam using an updated Lagrangian formulation (14). The structure is modeled with 40 elements providing 200 active DOF for Miller's solution and 240 for the current program. Displacement control was used with an initial displacement of 0.15m, 58 increments, a limit of 100 iterations per increment, and a tolerance of 0.5%. Figure 3-5 compares the tip displacement (w) solution of the current method to Miller's. Because this structure experiences large rotations, the current work should deviate from Miller's if the bending angle shape functions are the limiting factor of the previous work. However, the two solutions are almost identical. This is the first indication that the limit of the
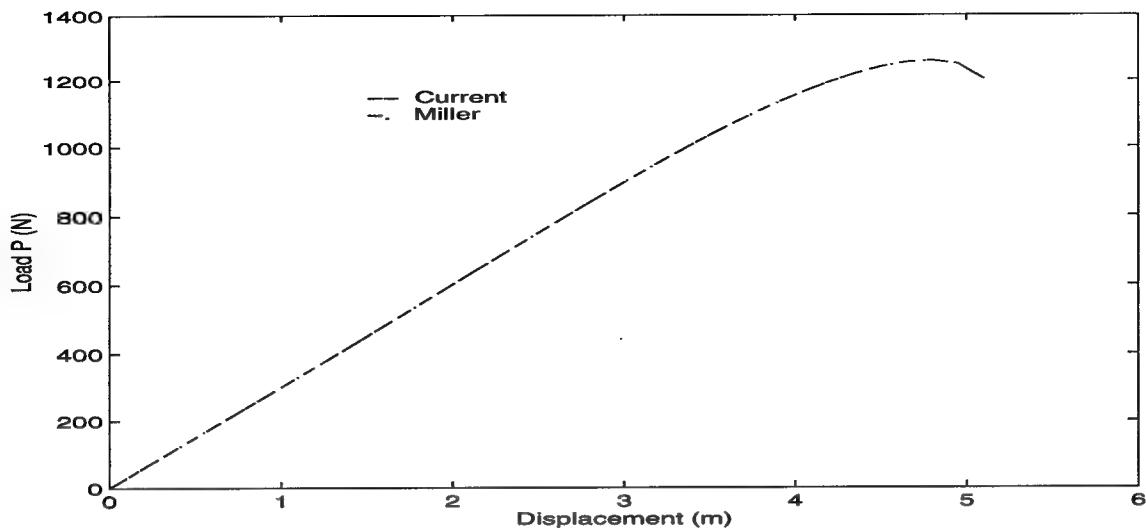
Figure 3-5 Cantilever Isotropic Beam End Displacement Comparison

previous effort is not in the finite element formulation, but in the derivation of the equilibrium equations.

The derivative of the bending angle ($\psi_{,2}$) is examined for the entire beam at maximum load. Figure 3-6 presents these results. As with the first example, the current effort generates a continuous curve
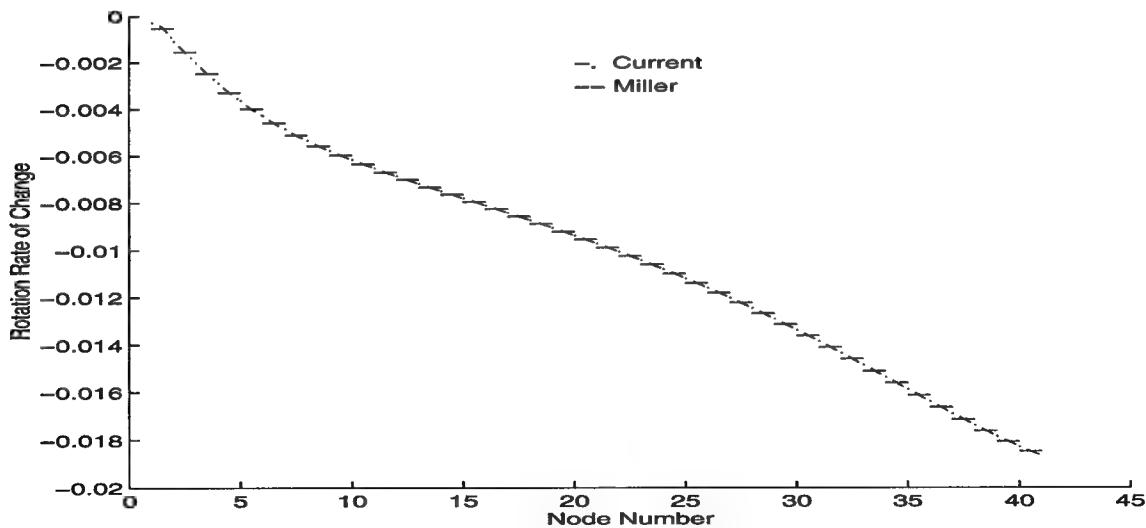
Figure 3-6 Cantilever Beam Rotation Rate of Change Comparison

whereas Miller's results are not. This indicates the Hermitian shape
functions do properly estimate the bending angle and its first
derivative.

The efficiency was also examined using the procedure outlined in
section 3.2.1. Miller's program required 14 elements and 70 active DOF
to match the solution generated using 40 elements. This model required
45.4 seconds to solve. The current work required fewer elements (13)
but more DOF (78). This caused the current program to run 70.7 seconds
before reaching a solution.

3.2.3 Very Deep Hinged Clamped Isotropic Arch. The following
example combines large rotations and large displacements. The
unsymmetrical boundary conditions combine with the very deep geometry,
as shown in figure 3-7, to present a formidable test of any large
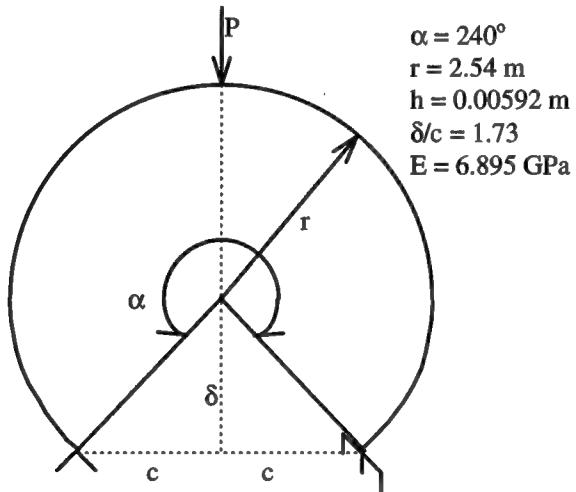
Figure 3-7 Very Deep Hinged Clamped Isotropic Arch

rotation and large displacement theory. The arch was modeled with 100
elements and the displacement control method was used. A maximum of 100
iterations was specified for each of the 200 increments, and the
tolerance was set at 2.0%.

The comparison with Miller's work is shown in Figure 3-8. The
programs produce nearly identical results up to the collapse load of
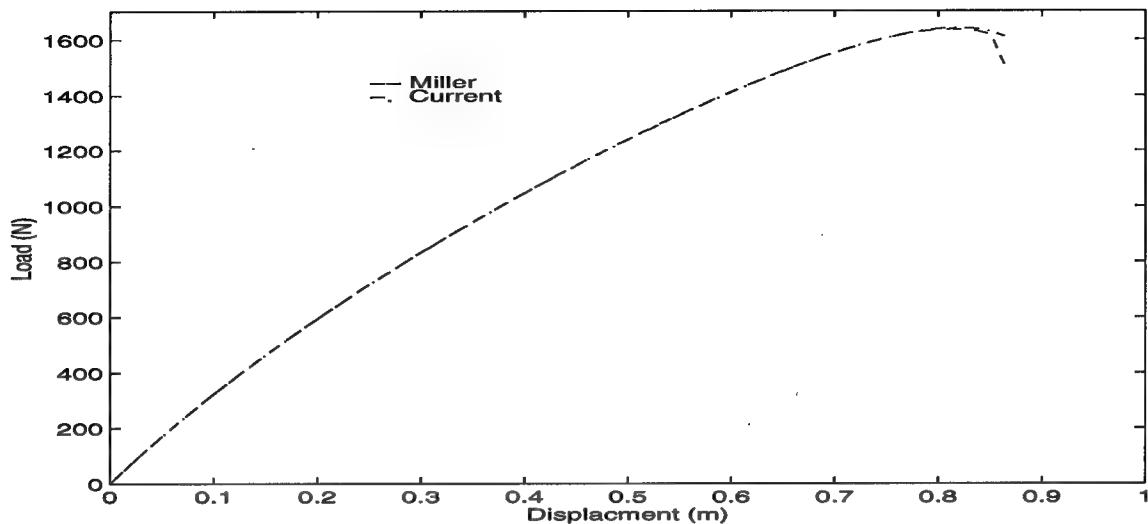
Figure 3-8 Load vs. Displacement for a Very Deep Hinged Clamped Arch

1.720 kn. and a displacement of 0.826 m. At this point the solutions
diverge. However, Miller discovered solutions generated after collapse
were unreliable due to large energy changes within the elements (13).
As such, solutions reached after collapse are neglected.

The rate of change of the bending rotation ($\psi_{,2}$) is presented in
figure 3-9 and shows the same results as the previous two examples.
Again, C^1 continuity of the bending angle is attained. Even with the
higher order continuity, the current model does not provide a better
solution. These results clearly indicate a limitation caused by the
linearization of the rotation angle (ψ) during derivation of the large
rotation kinematics.

Although the rotation limits have not been improved, the current
work does address another shortcoming of the previous efforts. At large
deflections, Miller noted a propensity for elements near the hinged
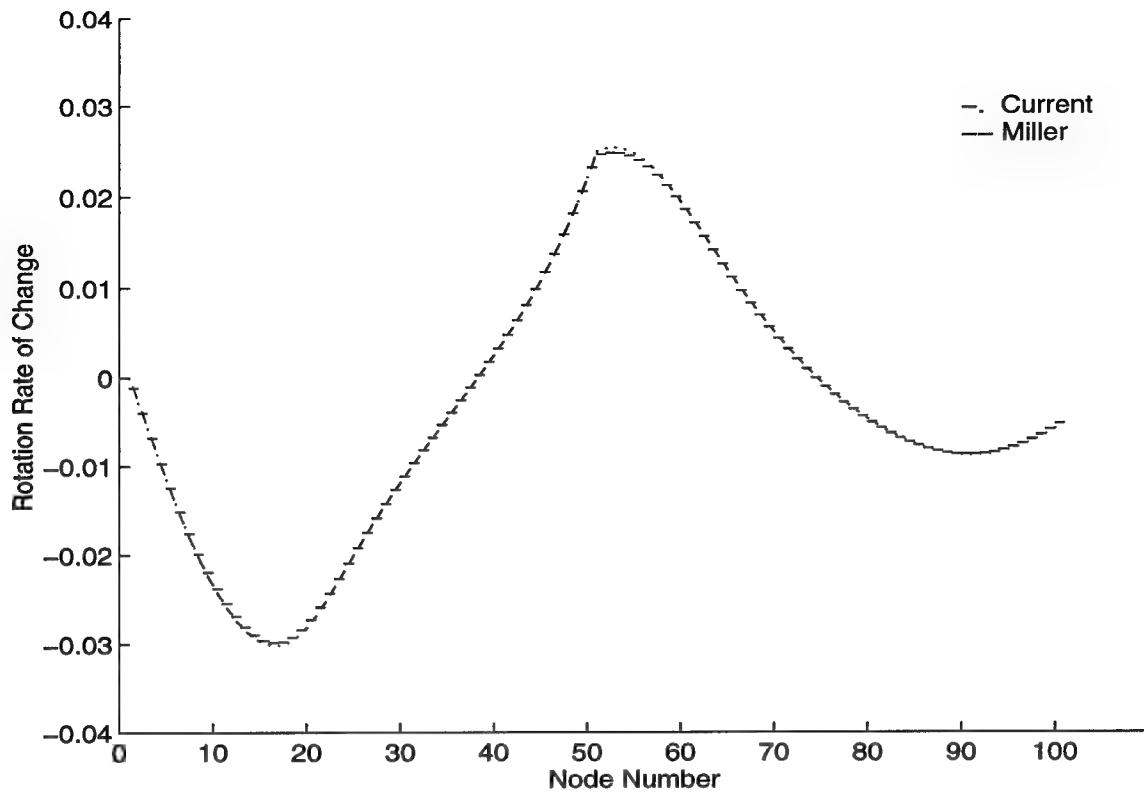
Figure 3-9 Rotation Rate of Change Comparison

boundary to "kink over" (13). Figure 3-10 shows the undeformed and

deformed shapes created by each model at 35 inches of vertical

deflection at the middle node. Despite providing similar solutions for

the displacement at the middle of the arch, the kinking phenomenon

Miller noticed is nonexistent in the current solution. As a result,

displacements near the hinged boundary are significantly different. For

this reason, the current model should be used for investigating large

rotation problems.

Another advantage of the current formulation is that erroneous

solutions are eliminated. In this example, Miller's formulation

continued providing solutions after the model experienced element
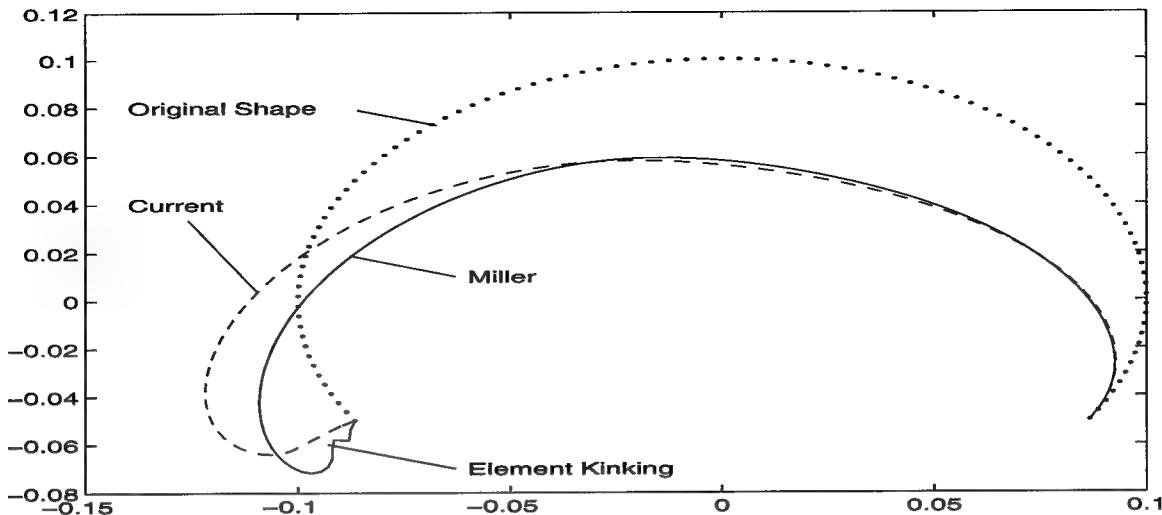
Figure 3-10 Deformed Shapes for Clamped Hinged Very Deep Arch

kinking despite the unstable nature of the structure (13). The current program does not converge on a solution beyond this deflection level.

The elimination of element kinking and inaccurate solutions clearly demonstrate the advantage of using Hermitian shape functions for interpolation of the bending angle (ψ) when modeling arches with large displacements and large rotations.

For this example, the current program required 98 elements with 586 active DOF to maintain an accurate solution. Miller's program only required 88 elements and 440 active DOF. As a result, the current formulation ran for 1049 seconds versus 805 seconds for the previous effort. Clearly, the trade off for C^1 continuity is a finer mesh and larger number of DOF, which in turn require more time to solve.

3.3 Arch Evaluation

This section makes use of the newly developed curved element to characterize certain geometric parameters. Two classes of composite arches are evaluated. The first group investigates how the arc

length (S) affects the collapse load (P_c). The second group evaluates the effect of changing the thickness ratio (h/r) versus P_c. In either case, simply supported composite arches are symmetrically modeled using the current program. All loads were applied and displacements (w) measured at the point of symmetry. The arches are constructed of a $[0/90]_s$ lay-up of AS4-3501-6 graphite epoxy. Each ply is 0.25mm thick, for a total thickness of 1mm. The lamina material properties are: E_1=142 GPa; E_2=9.8 GPa; G_{12}=G_{13}=6.0 GPa; G_{23}=4.8 GPa; and ν_{12}=0.3. These arches experience large deformations and moderate rotations. Riks method is used for all simulations with a tolerance of 0.5%, a maximum of 500 increments, and a limit of 100 iterations per increment. The initial load parameter (λ_0) was set at 0.1, with the number of iterations estimated as 2.5 (N_s), and a maximum allowable load change (λ_{max}) of 2.0.

3.3.1 Constant *h/r*, Variable Arc Length. For this portion of the evaluation, five different thickness ratios are considered. Each *h/r* was established by varying the radius while holding the thickness at 1mm. For each *h/r* the arc length was varied to determine how boundary location affects arch behavior. Table 3-1 presents the specifications of each arch analyzed for this section along with the depth and thickness parameters used for evaluation. Representative figures are presented in Appendix B.

Table 3-1 includes two classification parameters for thickness and depth evaluation. For thickness, either h/r or c/h may be used. The c/h parameter takes opening angle into account and is more accurate as the opening angle increases. Either δ/c or λ are used for depth classification. λ is calculated using $\lambda = [12(1-\upsilon^2)]^{.25}(r/h)^{.5}(\alpha/2)$.

Table 3-1 Arch Investigation Matrix, Constant h/r, $h = 0.001$m

h/r		S (m)	r (m)	c/h		δ/c		λ	
0.1	thick	0.01	0.01	4.79	thick	0.26	shallow	2.94	shallow
0.1	thick	0.015	0.01	6.82	thick	0.39	deep	4.41	shallow
0.1	thick	0.025	0.01	9.49	thick	0.72	deep	7.36	shallow
0.1	thick	0.05	0.01	5.98	thick	3.01	deep	14.71	deep
0.05	thick	0.015	0.02	7.33	thick	0.18	shallow	3.12	shallow
0.05	thick	0.02	0.02	9.59	thick	0.26	shallow	4.16	shallow
0.05	thick	0.025	0.02	11.70	thick	0.32	deep	5.20	shallow
0.05	thick	0.03	0.02	13.63	thick	0.39	deep	6.24	shallow
0.05	thick	0.05	0.02	19.98	thick	0.72	deep	10.40	deep
0.05	thick	0.075	0.02	19.08	thick	1.36	deep	15.61	deep
0.05	thick	0.1	0.02	11.97	thick	3.01	deep	20.81	deep
0.01	thin	0.025	0.1	12.47	thick	0.06	shallow	2.33	shallow
0.01	thin	0.05	0.1	24.74	thick	0.13	shallow	4.65	shallow
0.01	thin	0.075	0.1	36.63	thin	0.19	shallow	6.98	shallow
0.01	thin	0.1	0.1	47.94	thin	0.26	shallow	9.31	deep
0.01	thin	0.15	0.1	68.16	thin	0.39	deep	13.96	deep
0.01	thin	0.2	0.1	84.15	thin	0.55	deep	18.61	deep
0.005	thin	0.05	0.2	24.93	thick	0.06	shallow	3.29	shallow
0.005	thin	0.075	0.2	37.28	thin	0.09	shallow	4.93	shallow
0.005	thin	0.1	0.2	49.48	thin	0.13	shallow	6.58	shallow
0.005	thin	0.15	0.2	73.25	thin	0.19	shallow	9.87	deep
0.005	thin	0.2	0.2	95.89	thin	0.26	shallow	13.16	deep
0.005	thin	0.3	0.2	136.3	thin	0.39	deep	19.74	deep
0.0025	thin	0.05	0.4	24.98	thick	0.03	shallow	2.33	shallow
0.0025	thin	0.075	0.4	37.45	thin	0.05	shallow	3.49	shallow
0.0025	thin	0.1	0.4	49.87	thin	0.06	shallow	4.65	shallow
0.0025	thin	0.15	0.4	74.56	thin	0.09	shallow	6.98	shallow
0.0025	thin	0.2	0.4	98.96	thin	0.13	shallow	9.31	deep
0.0025	thin	0.4	0.4	191.8	thin	0.26	shallow	18.61	deep
0.0025	thin	0.6	0.4	272.6	thin	0.29	deep	27.92	deep

Similar to the thickness classification parameters, as α increases, the λ parameter is preferred because it directly accounts for the opening angle.

Figures 3-11 presents a typical P_c vs. w curve. The critical load is defined as the local peak indicated by point A. In the cases where no local peak exists, it is assumed the structure does not collapse but has post buckling features. Figure 3-12 presents the P_c v S data for $h/r = 0.1$. As arc length approaches zero, P_c smoothly approaches
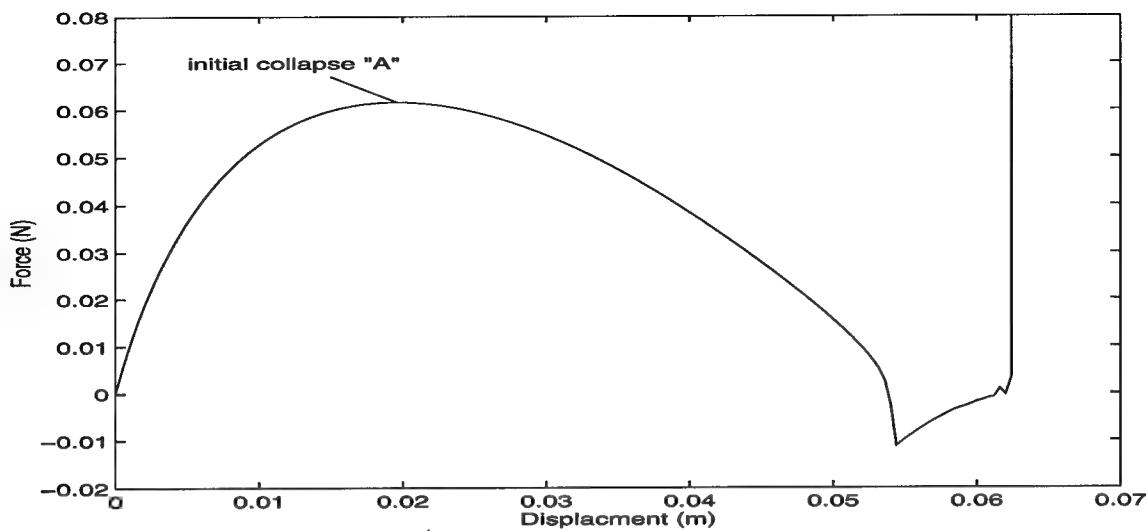
Figure 3-11 Typical Arch P_c vs. w Curve
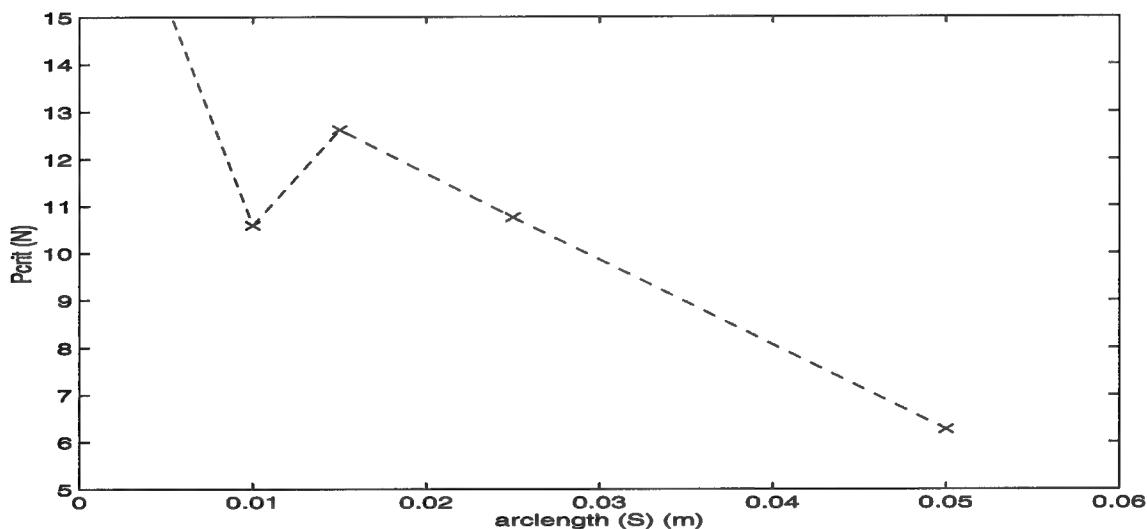
Figure 3-12 Critical Load Comparison for $h/r = 0.1$

infinity except for the decrease at $S = 0.01$m. Further examination indicates this arch experiences local snapping and does not totally collapse (total collapse is defined as global snapping). That is, the arch geometry changes but is immediately able to support a load with

minimal deflection increases. Figure 3-13 presents the load

displacement curve for an arch with arc length of 0.01m and thickness

ratio of 0.1. As indicated, the arch initially collapses at 10.059 N at
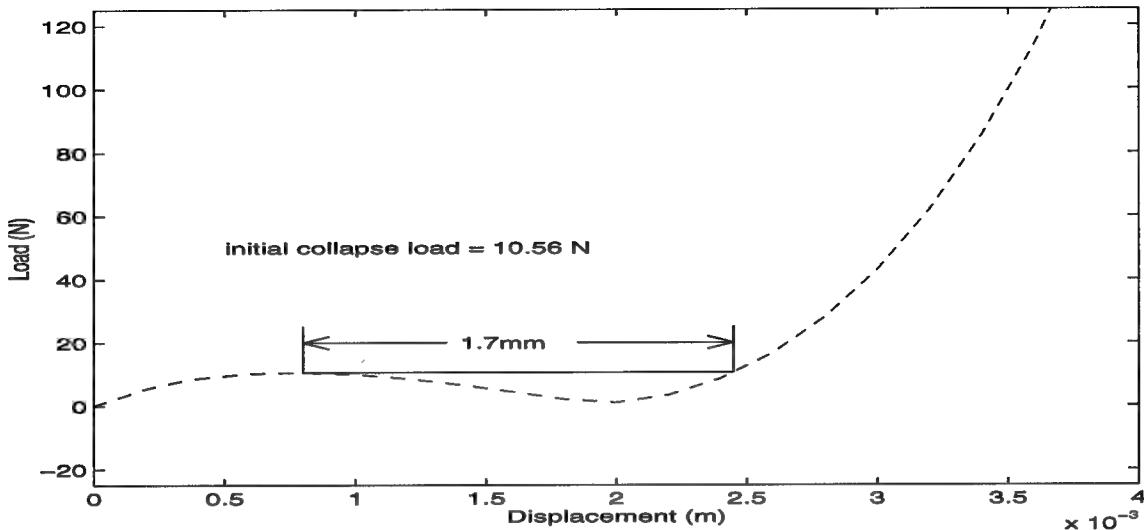
Figure 3-13 Load v Displacement for $S = 0.01$m, $h/r = 0.1$

a displacement of 0.75mm. This arch is able tosupport the same load

again after the displacement increases only 1.7mm to 2.45mm. Figure 3-

14 presents the actual arch shape at the collapse displacement. The

counterflexure points indicate the point on the arch where the radius

changes signs. Outside these points, the arch isconcave downward and

between the counterflexure points the arch is concave upward. The

distance between the points is indicative of the stability the arch. As

the distance increases, stability decreases along with the ability to

support a load. As indicated in figure 3-14, the counterflexure points

of the $S = 0.01$m, $h/r = 0.1$ arch are separated by 1.5mm or 15% of the arc

length. This small separation, combined with the minimal displacement

change, indicate that a local snapping occurs.
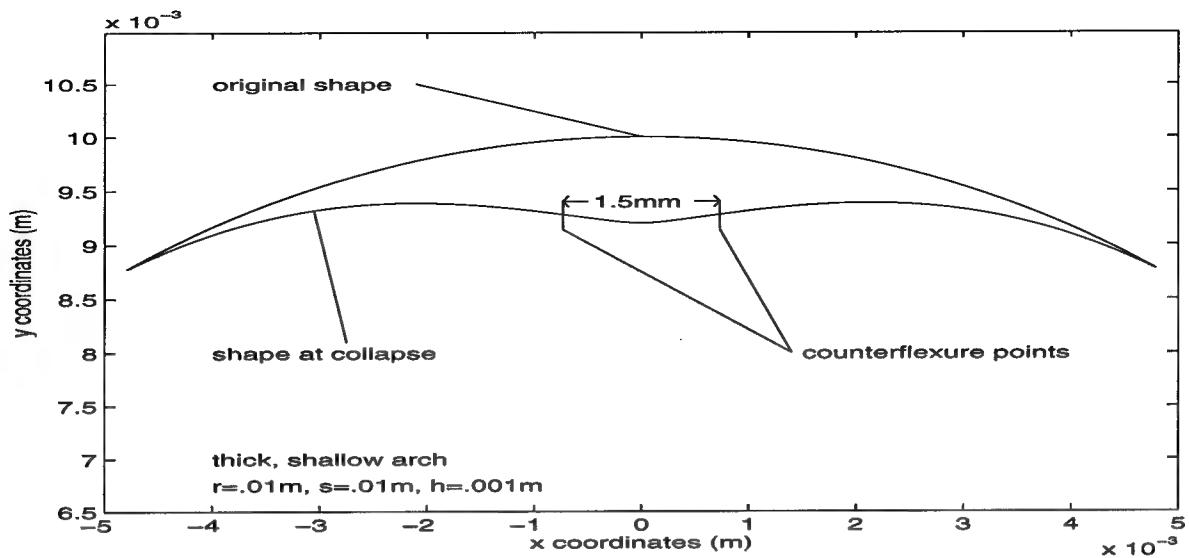
Figure 3-14 Undeformed and Collapsed Arch Shapes; $S = 0.01$m, $h/r = 0.1$

To demonstrate the total collapse of an arch, the typical load displacement curve for an arch with $S = 0.025$m and $h/r = 0.1$ is considered. This curve is presented in figure 3-15. This structure initially collapses at a load of 10.76N and a displacement of 2.72mm.
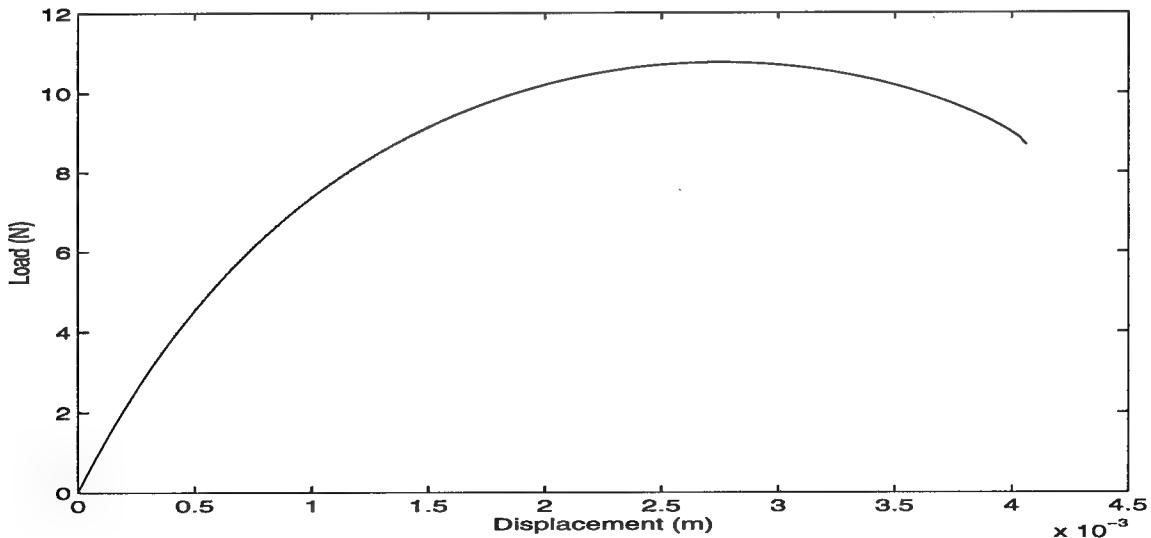
Figure 3-15 Load v Displacement for $S = 0.025$m, $h/r = 0.1$

As shown, the arch does not recover after the initial collapse point, indicating a global snapping or total collapse of the structure.

The undeformed shape of this arch is presented in figure 3-16 along with the shape at collapse. The counterflexure points are
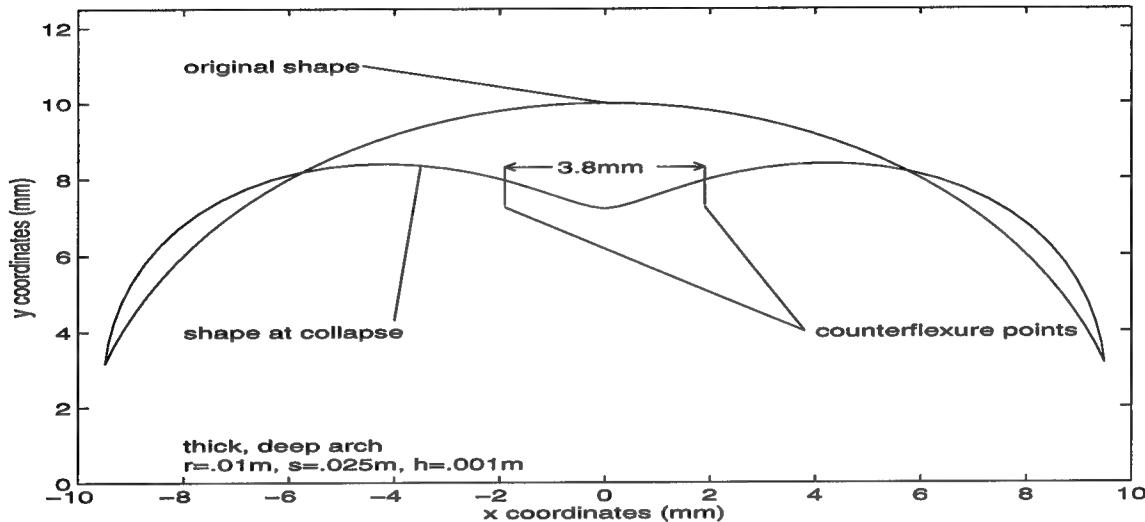
Figure 3-16 Undeformed and Collapsed Arch Shapes; $S = 0.025$m, $h/r = 0.1$

separated by 3.8mm (15.2% of S) at collapse. Both this arch ($S = 0.025$, $h/r = 0.1$) and the previous arch ($S = 0.01$, $h/r = 0.1$) have similar counterflexure separation distances (15% of S) and collapse loads (10.5N).

Although both arches collapse at similar loads, the longer arch displaces almost 12% of its length before collapsing versus 7% for the shorter arch. Also, the longer arch does not recover after initial collapse, indicating a that global snapping occurs.

Similar results were observed for thickness ratios (h/r) of 0.05, 0.01, 0.005, and 0.0025. These results are summarized in table 3-2 and presented graphically in Appendix B.

Table 3-2 Constant *h/r* Results

h/r	S (m)	P_c (N)	w_c (mm)	recover	post snap disp (mm)	type of snapping
0.1	0.01	10.587	0.7471	yes	1.70292	local
0.1	0.015	12.616	1.2727	no	n/a	global
0.1	0.025	10.761	2.7196	no	n/a	global
0.1	0.05	6.2709	7.059	no	n/a	global
0.05	0.015	4.8981	0.8508	yes	1.9892	local
0.05	0.02	5.8055	1.2548	no	n/a	global
0.05	0.025	5.6238	1.7891	no	n/a	global
0.05	0.03	5.1597	2.385	no	n/a	global
0.05	0.05	3.5331	5.7071	no	n/a	global
0.05	0.075	2.7036	12.055	no	n/a	global
0.05	0.1	1.7821	14.187	no	n/a	global
0.01	0.025	0.3848	0.66	yes	0.645	local
0.01	0.05	0.67085	1.6038	yes	0.808	local
0.01	0.075	0.52102	3.23	no	n/a	global
0.01	0.1	0.40145	5.5168	no	n/a	global
0.01	0.15	0.26823	11.795	no	n/a	global
0.01	0.2	0.19992	19.769	no	n/a	global
0.005	0.05	0.20175	0.97761	yes	2.31239	local
0.005	0.075	0.2405	1.81	yes	5.6552	local
0.005	0.1	0.1984	2.9864	no	n/a	global
0.005	0.15	0.13688	6.4393	no	n/a	global
0.005	0.2	0.10279	10.941	no	n/a	global
0.005	0.3	0.06824	23.472	no	n/a	global
0.0025	0.05	0.04979	0.6667	yes	0.6532	local
0.0025	0.075	0.07853	0.9867	yes	2.8736	local
0.0025	0.1	0.08843	1.654	yes	4.7291	local
0.0025	0.15	0.06757	3.366	no	n/a	global
0.0025	0.2	0.05178	5.829	no	n/a	global
0.0025	0.4	0.02605	21.968	no	n/a	global
0.0025	0.6	0.0133	47.795	no	n/a	global

The results presented in table 3-2 highlight two important trends in arch behavior. First, the arches that recover after collapse, or experience local snapping, are generally classified as shallow structures, based on the specifications listed in table 3-1. It is interesting to note that for smaller radii (r =0.01m and 0.02m), the δ/c parameter correctly classifies shallow arches as those that experience local snapping, but as the radius increases, the λ parameter provides a more accurate prediction.

The second trend is for P_c to increase as h/r increases for a given arc length. This indicates the influence of through the thickness shear, as will be described in the next section.

3.3.2 Constant Arc Length, Variable h/r. Table 3-3 presents the specifications of each arch analyzed for this section along with the depth and thickness parameters used for evaluation.

Table 3-3 Arch Investigation Matrix, Constant S, $h = 0.001$m

S (m)	h/r		r (m)	c/h		δ/c		λ	
0.025	0.1	thick	0.01	9.49	thick	0.72	deep	7.36	shallow
0.025	0.05	thick	0.02	11.70	thick	0.32	deep	5.20	shallow
0.025	0.01	thin	0.1	12.47	thick	0.06	shallow	2.33	shallow
0.05	0.1	thick	0.01	5.98	thick	3.01	deep	14.71	deep
0.05	0.05	thick	0.02	18.98	thick	0.72	deep	10.40	deep
0.05	0.01	thin	0.1	24.74	thick	0.13	shallow	4.65	shallow
0.05	0.005	thin	0.2	24.93	thick	0.06	shallow	3.29	shallow
0.075	0.05	thick	0.02	19.08	thick	1.36	deep	15.61	deep
0.075	0.01	thin	0.1	36.63	thin	0.19	shallow	6.98	shallow
0.075	0.005	thin	0.2	37.28	thin	0.09	shallow	4.93	shallow
0.075	0.0025	thin	0.4	37.45	thin	0.05	shallow	3.49	shallow
0.1	0.05	thick	0.02	11.97	thick	3.01	deep	20.81	deep
0.1	0.01	thin	0.1	47.94	thin	0.26	shallow	9.31	deep
0.1	0.005	thin	0.2	49.48	thin	0.13	shallow	6.58	shallow
0.1	0.0025	thin	0.4	49.87	thin	0.06	shallow	4.65	shallow
0.15	0.01	thin	0.1	68.16	thin	0.39	deep	13.96	deep
0.15	0.005	thin	0.2	73.25	thin	0.19	shallow	9.87	deep
0.15	0.0025	thin	0.4	74.56	thin	0.09	shallow	6.97	shallow
0.2	0.01	thin	0.1	84.15	thin	0.55	deep	18.61	deep
0.2	0.005	thin	0.2	95.89	thin	0.26	shallow	13.16	deep
0.2	0.0025	thin	0.4	98.96	thin	0.13	shallow	9.31	deep

Figure 3-17 presents a P_c vs. h/r curve for arches 0.025m long. This curve was constructed by varying the radius of curvature while maintaining a thickness of 1mm. As shown in figure 3-18, as h/r decreases due to the increasing radius, the depth of the arch decreases or becomes more like a flat beam.

3-17
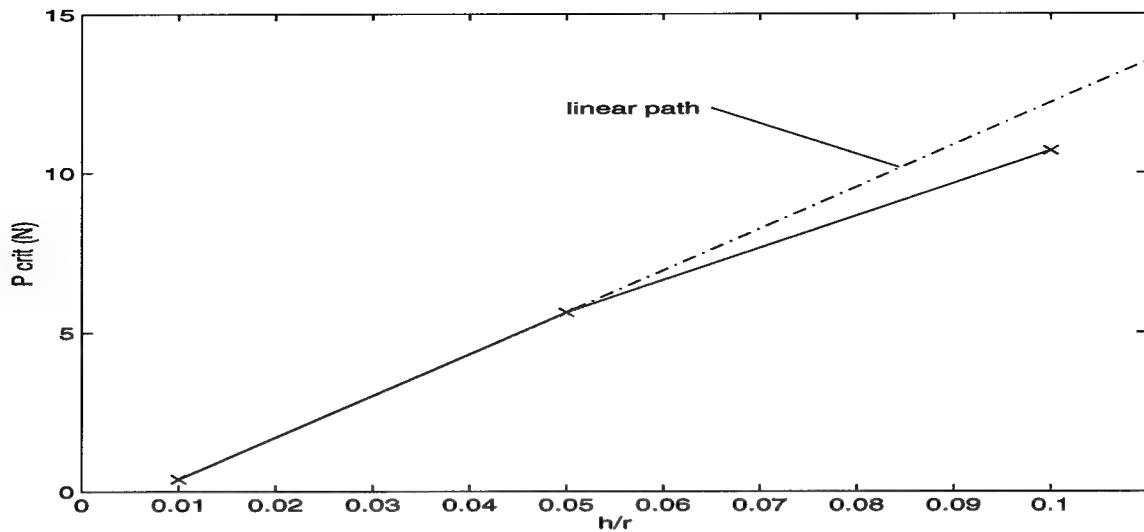
Figure 3-17 P_c vs. *h/r* for $S = 0.025$m

r_1

$r_1 < r_2 < r_3$

r_2

r_3
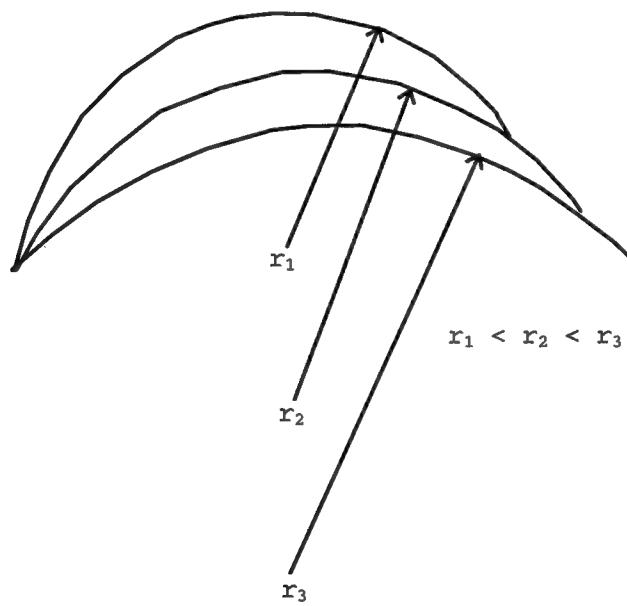
Figure 3-18 Effects of Increasing Radius for a Constant Arc Length

As expected, P_c increases as h/r increases; however, as indicated in figure 3-17, the rate of change of the critical load decreases as the thickness ratio increases (radius decreases). All three arches used to

3-18

generate the curve are considered thick based on $c/h < 25$. As such, the deviations observed in the P_c vs. h/r curve demonstrate the nonlinear effects caused by significant through the thickness shear stresses experienced by thick beams and arches.

Figure 3-19 presents the P_c vs. h/r for an arc length of 0.2m. A slight variation in the critical load rate of change was observed. In this case, all three $S = 0.2m$ arches were thin arches, based on the $c/h > 25$ criteria presented in table 3-3. As such, the P_c vs. h/r curve
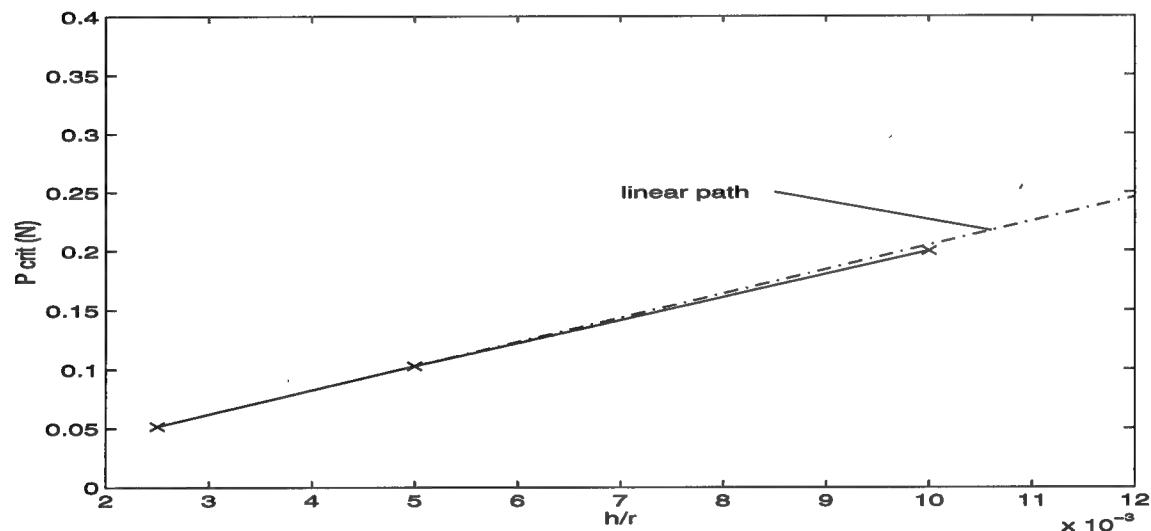
Figure 3-19 P_c vs. h/r for $S = 0.2m$

indicates minimal nonlinear behavior caused by through the thickness shear.

The critical displacement (w_c) for these arches was also examined. Figure 3-20 shows the w_c vs. h/r curve for $S = 0.025m$. This curve deviates from the linear path as indicated. Figure 3-21 presents the w_c vs. h/r curve for $S = 0.2m$. In both cases, as the arches modeled become

deeper, the rate of change of the critical displacement decreases, irregardless of the presence of through the thickness shear effects.

The analysis of arches with constant arc length and thickness and changing radius of curvature demonstrates the nonlinearities introduced as depth and through the thickness shear increase.

Similar P_c vs. h/r and w_c vs. h/r curves were constructed for arc lengths of 0.05m, 0.075m, 0.1m, and 0.15m. These curves further exemplify the relationships between critical load and thickness and critical displacement and depth. Appendix C contains these curves.
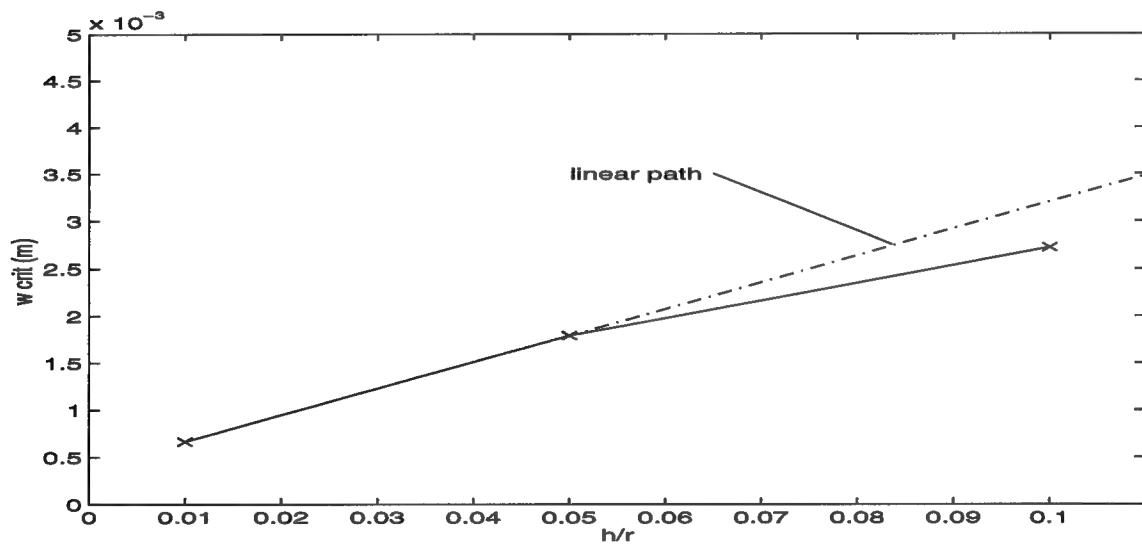
Figure 3-20 w_c vs. h/r for S = 0.025m
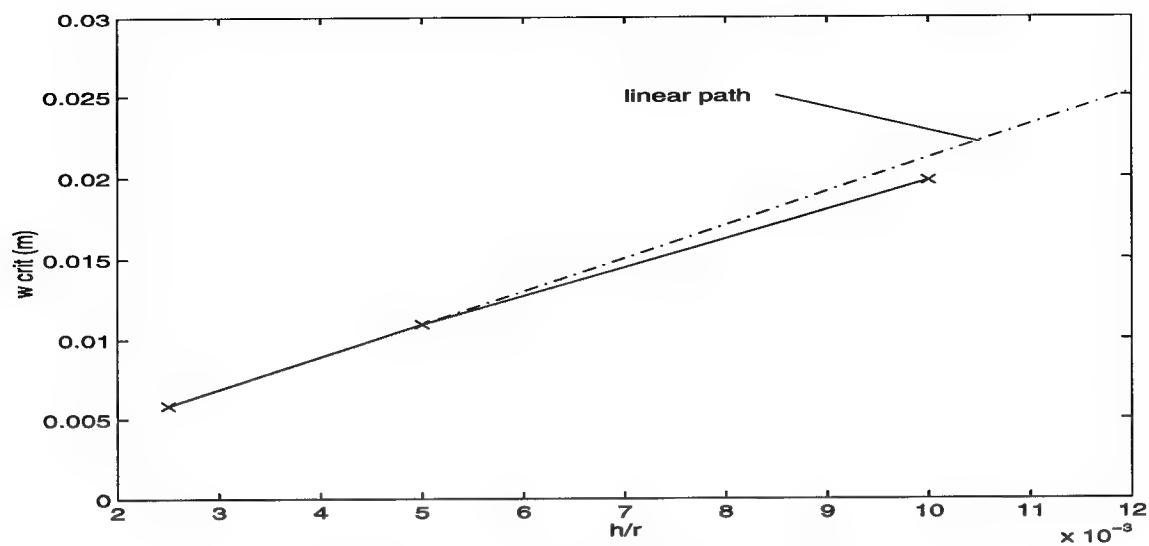
Figure 3-21 w_c vs. h/r for $S = 0.2$m

IV. Conclusions and Recommendations

4.1 Summary and Conclusions

This effort successfully modified the existing finite element formulation of a potential energy based large deformation and moderate rotation theory. Comparison with the unmodified formulation indicated proper incorporation of Hermitian shape functions for the bending angle (ψ). Despite the higher order shape functions, the rotation limit of the current work remained the same as the previous effort. Although the limits were unchanged, the current formulation did eliminate element kinking experienced in previous efforts.

Numerous arch configurations were modeled to analyze their behavior with respect to arc length and thickness variations. As the arc length decreased for a given thickness ratio (h/r), the critical load (P_c) approached infinity except for a minor decrease as the arch became flatter. This dip in the P_c vs. S curve comes about because global snapping becomes a local phenomena before snapping disappears altogether. The decrease in critical load does not represent a structural failure, but a geometry change which does not affect the load carrying capability of the arch. An arch experiencing local snapping recovers its structural integrity with only a small displacement increase. On the other hand, after global snapping, an arch experiences extremely large displacements before recovering. The extent of snapping (local or global) can be accurately predicted by examining the depth of an arch. Deep arches experience total collapse while shallow arches only undergo local snapping.

Varying the radius of curvature of an arch while holding the thickness and arc length constant demonstrated independence of depth and thickness. Compared to an analysis where through the thickness shear is neglected, these stresses decreased the critical load of both shallow and deep arches as the thickness ratio increased. Similarly, as the depth increased, nonlinearities decreased critical displacement for thick and thin beams alike. Thus, the critical load is dependent on the thickness and the critical displacement is a function of the depth of the arch.

4.2 Recommendations

The current model is limited to 45 degrees of bending rotation just as the previous effort was. This indicates the previous finite element formulation was sufficient and the underlying equilibrium equations are limiting the accuracy of the program beyond the bending rotation limits. As such, the equilibrium equations should be rederived taking the first variation of the potential energy equation with respect to the bending angle (ψ) as well as vertical displacement (w). Such a derivation could then be used with the current finite element formulation to increase bending angle limit.

A.1 Q_{ij} Transformations of Eqn(2-18)

$$\overline{Q}_{11} = Q_{11}\cos^4\theta + 2(Q_{12}+2Q_{66})\sin^2\theta\cos^2\theta + Q_{22}\sin^4\theta$$

$$\overline{Q}_{12} = (Q_{11}+Q_{22}-4Q_{66})\sin^2\theta\cos^2\theta + Q_{12}(\cos^4\theta+\sin^4\theta)$$

$$\overline{Q}_{22} = Q_{11}\sin^4\theta + 2(Q_{12}+2Q_{66})\sin^2\theta\cos^2\theta + Q_{22}\cos^4\theta$$

$$\overline{Q}_{16} = (Q_{11}-Q_{12}-2Q_{66})\sin\theta\cos^3\theta + (Q_{12}-Q_{22}+2Q_{66})\sin^3\theta\cos\theta$$

$$\overline{Q}_{26} = (Q_{11}-Q_{12}-2Q_{66})\sin^3\theta\cos\theta + (Q_{12}-Q_{22}+2Q_{66})\sin\theta\cos^3\theta$$

$$\overline{Q}_{66} = (Q_{11}+Q_{22}-2Q_{12}-2Q_{66})\sin^2\theta\cos^2\theta + Q_{66}(\cos^4\theta+\sin^4\theta)$$

$$\overline{Q}_{44} = Q_{55}\sin^2\theta + Q_{44}\cos^2\theta$$

$$\overline{Q}_{45} = (Q_{44}-Q_{55})\cos\theta\sin\theta$$

$$\overline{Q}_{55} = Q_{44}\sin^2\theta + Q_{55}\cos^2\theta$$

A.2 L, S, and H Matrices in Eqn(2-44) and Eqn(2-46)

$$L_o^T = \{0 \quad 1 \quad -c \quad 0 \quad 0 \quad 0 \quad 0\}$$

$$L_1^T = \{0 \quad 0 \quad -c^2 \quad 0 \quad 0 \quad 0 \quad 1\}$$

$$L_2^T = \{0 \quad -c^2 \quad 0 \quad 0 \quad 0 \quad 0 \quad c\}$$

$$L_3^T = \{0 \quad 0 \quad 0 \quad 0 \quad k \quad 0 \quad k\}$$

$$L_4^T = \{0 \quad 0 \quad 0 \quad 0 \quad ck \quad 0 \quad ck\}$$

$$L_5^T = \{0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\}$$

$$L_6^T = \{0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\}$$

$$S_1^T = \{0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\}$$

$$S_2^T = \{0 \quad 0 \quad 0 \quad 3k \quad 0 \quad 3k \quad 0\}$$

$$H_0 = \begin{bmatrix} c^2 & 0 & 0 & c & 0 & 0 & 0 \\ 0 & 1 & -c & 0 & 0 & 0 & 0 \\ 0 & -c & c^2 & 0 & 0 & 0 & 0 \\ c & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$H_1 = \begin{bmatrix} 0 & 0 & 0 & c^2 & 0 & c^2 & 0 \\ 0 & 0 & -c^2 & 0 & 0 & 0 & 1+\psi^2 \\ 0 & -c^2 & 2c^3 & 0 & 0 & 0 & -c(1+\psi^2) \\ c^2 & 0 & 0 & 2c & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ c^2 & 0 & 0 & c & 0 & 0 & \psi \\ 0 & 1+\psi^2 & -c(1+\psi^2) & 0 & 0 & \psi & 0 \end{bmatrix}$$

$$H_2 = \begin{bmatrix} -3c^4 & 0 & 0 & -2c^3 & 0 & c^3 & 0 \\ 0 & -3c^2 & 2c^3 & 0 & 0 & 0 & c(1+\psi^2) \\ 0 & 2c^3 & 0 & 0 & 0 & 0 & -2c^2 \\ -2c^3 & 0 & 0 & 0 & 0 & 2c^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ c^3 & 0 & 0 & 2c^2 & 0 & c^2 & c\psi \\ 0 & c(1+\psi^2) & -2c^2 & 0 & 0 & c\psi & 1+2\psi^2 \end{bmatrix}$$

$$H_3 = \begin{bmatrix} 2c^5 & 0 & 0 & kc^2 & 0 & -2c^4+kc^2 & 0 \\ 0 & 2c^3 & 0 & 0 & k & 0 & -2c^2+k \\ 0 & 0 & 0 & 0 & -kc & 0 & -kc \\ kc^2 & 0 & 0 & 2kc & 0 & kc & 0 \\ 0 & k & -kc & 0 & 0 & 0 & 0 \\ -2c^4+kc^2 & 0 & 0 & kc & 0 & 2c^3 & 0 \\ 0 & -2c^2+k & -kc & 0 & 0 & 0 & 2c \end{bmatrix}$$

$$H_4 = \begin{bmatrix} 0 & 0 & 0 & kc^3 & 0 & kc^3 & 0 \\ 0 & 0 & 0 & 0 & kc & 0 & kc \\ 0 & 0 & 0 & 0 & -2kc^2 & 0 & -2kc^2 \\ kc^3 & 0 & 0 & 4kc^2 & 0 & 3kc^2 & 0 \\ 0 & kc & -2kc^2 & 0 & 0 & 0 & k(1+\psi^2) \\ kc^3 & 0 & 0 & 3kc^2 & 0 & 2kc^2 & 0 \\ 0 & kc & -2kc^2 & 0 & k(1+\psi^2) & 0 & 2k(1+\psi^2) \end{bmatrix}$$

$$H_5 = \begin{bmatrix} 0 & 0 & 0 & -2kc^4 & 0 & -2kc^4 & 0 \\ 0 & 0 & 0 & 0 & -2kc^2 & 0 & -2kc^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2kc^4 & 0 & 0 & 0 & 0 & 2kc^3 & 0 \\ 0 & -2kc^2 & 0 & 0 & 0 & 0 & 2kc(1+\psi^2) \\ -2kc^4 & 0 & 0 & -2kc^3 & 0 & 4kc^3 & 0 \\ 0 & -2kc^2 & 0 & 0 & 2kc(1+\psi^2) & 0 & 4kc \end{bmatrix}$$

$$H_6 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & k^2c^2 & 0 & k^2c^2 & 0 \\ 0 & 0 & 0 & 0 & k^2 & 0 & k^2 \\ 0 & 0 & 0 & k^2c^2 & 0 & k^2c^2 & 0 \\ 0 & 0 & 0 & 0 & k^2 & 0 & k^2 \end{bmatrix}$$

$$H_7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2k^2c^3 & 0 & 2k^2c^3 & 0 \\ 0 & 0 & 0 & 0 & 2k^2c & 0 & 2k^2c \\ 0 & 0 & 0 & 2k^2c^3 & 0 & 2k^2c^3 & 0 \\ 0 & 0 & 0 & 0 & 2k^2c & 0 & 2k^2c \end{bmatrix}$$

A.3 Element Stiffness Matrices of Eqn(2-47)

$$\hat{K} = AL_0 L_0^T + D(L_0 L_2^T + L_2 L_0^T + L_1 L_1^T)$$
$$\quad + F(L_0 L_4^T + L_4 L_0^T + L_1 L_3^T + L_3 L_1^T + L_2 L_2^T)$$
$$\quad + H(L_2 L_4^T + L_4 L_2^T + L_3 L_3^T) + JL_4 L_4^T + ASS_0 S_0^T$$
$$\quad + DS(S_0 S_2^T + S_2 S_0^T +) + FSS_2 S_2^T$$

$$\hat{N}_1 = A(L_0 d^T H_0 + d^T L_0 H_0 + H_0 dL_0^T)$$
$$\quad + D(L_0 d^T H_2 + L_2 d^T H_0 + L_1 d^T H_1 + d^T L_0 H_2 + d^T L_2 H_0 + d^T L_1 H_1 + H_2 dL_0^T + H_0 dL_2^T + H_1 dL_1^T)$$
$$\quad + F\left(\begin{matrix} L_0 d^T H_4 + L_4 d^T H_0 + L_1 d^T H_3 + L_3 d^T H_1 + L_2 d^T H_2 + d^T L_0 H_4 + d^T L_4 H_0 + d^T L_1 H_3 \\ + d^T L_3 H_1 + d^T L_2 H_2 + H_4 dL_0^T + H_0 dL_4^T + H_3 dL_1^T + H_1 dL_3^T + H_2 dL_2^T \end{matrix} \right)$$
$$\quad + H\left(\begin{matrix} L_0 d^T H_6 + L_1 d^T H_5 + L_2 d^T H_4 + L_3 d^T H_3 + L_4 d^T H_2 + d^T L_0 H_6 + d^T L_1 H_5 + d^T L_2 H_4 \\ + d^T L_3 H_3 + d^T L_4 H_2 + H_6 dL_0^T + H_5 dL_1^T + H_4 dL_2^T + H_3 dL_3^T + H_2 dL_4^T \end{matrix} \right)$$
$$\quad + J\left(\begin{matrix} L_1 d^T H_7 + L_2 d^T H_6 + L_3 d^T H_5 + L_4 d^T H_4 + d^T L_1 H_7 + d^T L_2 H_6 + d^T L_3 H_5 + d^T L_4 H_4 \\ + H_7 dL_1^T + H_6 dL_2^T + H_5 dL_3^T + H_4 dL_4^T \end{matrix} \right)$$
$$\quad + L(L_3 d^T H_7 + L_4 d^T H_4 + d^T L_3 H_3 + d^T L_4 H_4 + H_7 dL_3^T + H_4 dL_4^T)$$

$$\hat{N}_2 = A\left(H_0 dd^T H_0 + \frac{1}{2} d^T H_0 dH_0\right)$$

$$+ D\left(\begin{array}{l} H_0 dd^T H_2 + \frac{1}{2} H_2 dd^T H_0 + H_1 dd^T H_1 \\ + \frac{1}{4} d^T H_0 dH_2 + \frac{1}{4} d^T H_2 dH_0 + \frac{1}{2} d^T H_1 dH_1 \end{array}\right)$$

$$+ F\left(\begin{array}{l} H_0 dd^T H_4 + \frac{1}{2} H_4 dd^T H_0 + H_1 dd^T H_3 + H_3 dd^T H_1 + H_2 dd^T H_2 \\ + \frac{1}{4} d^T H_0 dH_4 + \frac{1}{4} d^T H_4 dH_0 + \frac{1}{2} d^T H_1 dH_3 + \frac{1}{2} d^T H_3 dH_1 + \frac{1}{2} d^T H_2 dH_2 \end{array}\right)$$

$$+ H\left(\begin{array}{l} \frac{1}{2} H_0 dd^T H_6 + \frac{1}{2} H_6 dd^T H_0 + H_1 dd^T H_5 + H_5 dd^T H_1 + H_2 dd^T H_4 + H_4 dd^T H_2 + H_3 dd^T H_3 \\ + \frac{1}{4} d^T H_0 dH_6 + \frac{1}{4} d^T H_6 dH_0 + \frac{1}{2} d^T H_1 dH_5 + \frac{1}{2} d^T H_5 dH_1 + \frac{1}{2} d^T H_2 dH_4 + \frac{1}{2} d^T H_4 dH_2 + \frac{1}{2} d^T H_3 dH_3 \end{array}\right)$$

$$+ J\left(\begin{array}{l} H_1 dd^T H_7 + H_7 dd^T H_1 + H_2 dd^T H_6 + H_6 dd^T H_2 + H_3 dd^T H_5 + H_5 dd^T H_3 + H_4 dd^T H_4 \\ + \frac{1}{2} d^T H_1 dH_7 + \frac{1}{2} d^T H_7 dH_1 + \frac{1}{2} d^T H_2 dH_6 + \frac{1}{2} d^T H_6 dH_2 + \frac{1}{2} d^T H_3 dH_5 + \frac{1}{2} d^T H_5 dH_3 + \frac{1}{2} d^T H_4 dH_4 \end{array}\right)$$

$$+ L\left(\begin{array}{l} H_3 dd^T H_7 + H_7 dd^T H_3 + H_4 dd^T H_6 + H_6 dd^T H_4 + H_5 dd^T H_5 \\ + \frac{1}{2} d^T H_3 dH_7 + \frac{1}{2} d^T H_7 dH_3 + \frac{1}{2} d^T H_4 dH_6 + \frac{1}{2} d^T H_6 dH_4 + \frac{1}{2} d^T H_5 dH_5 \end{array}\right)$$

$$+ R\left(\begin{array}{l} H_5 dd^T H_7 + H_7 dd^T H_5 + H_6 dd^T H_6 \\ + \frac{1}{2} d^T H_5 dH_7 + \frac{1}{2} d^T H_7 dH_5 + \frac{1}{2} d^T H_6 dH_6 \end{array}\right)$$

$$+ T\left(H_7 dd^T H_7 + \frac{1}{2} d^T H_7 dH_7\right)$$
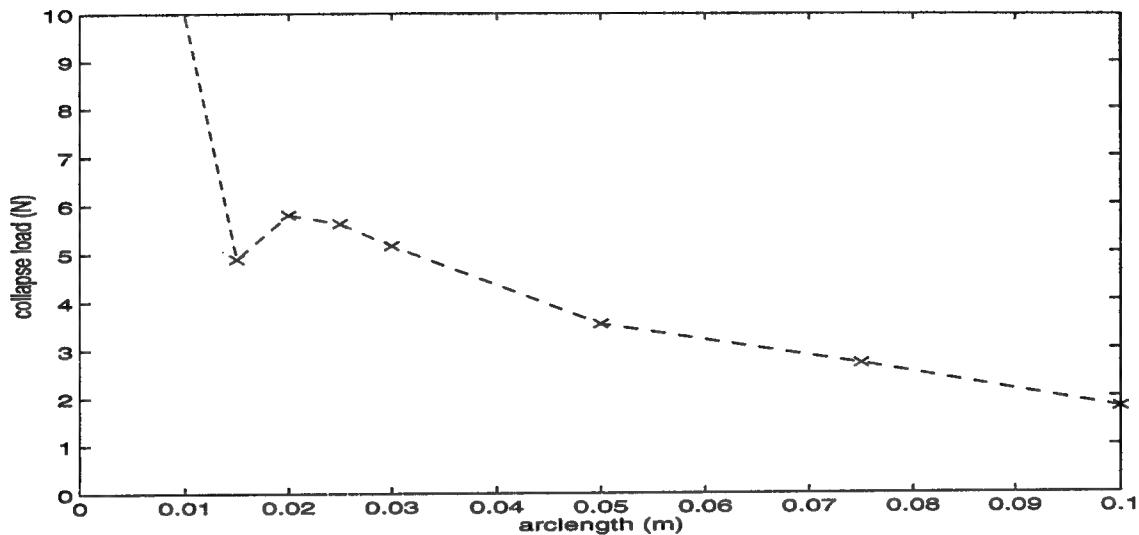
Figure B-1 Critical Load Comparison for $h/r = 0.05$
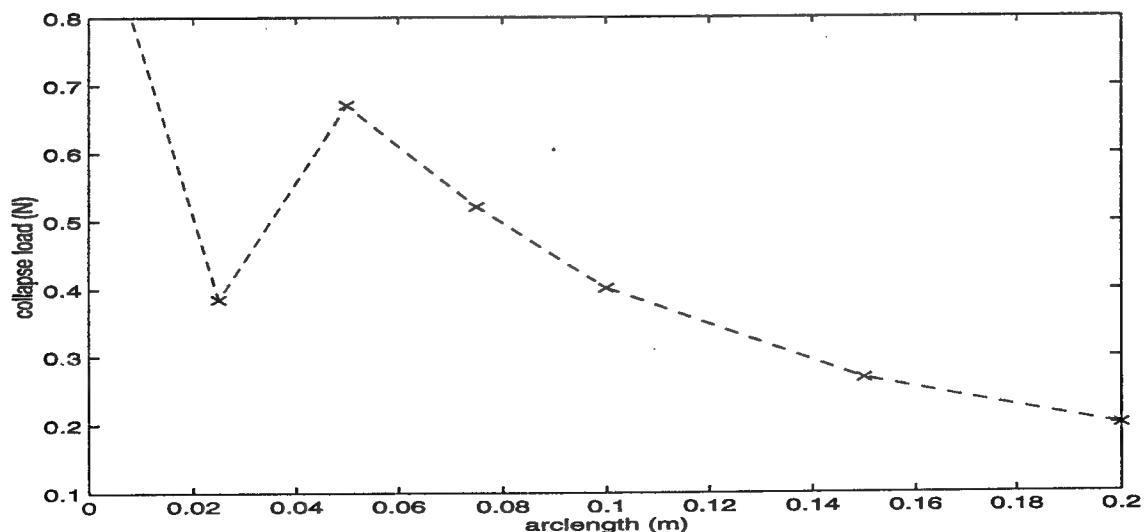
Figure B-2 Critical Load Comparison for $h/r = 0.01$
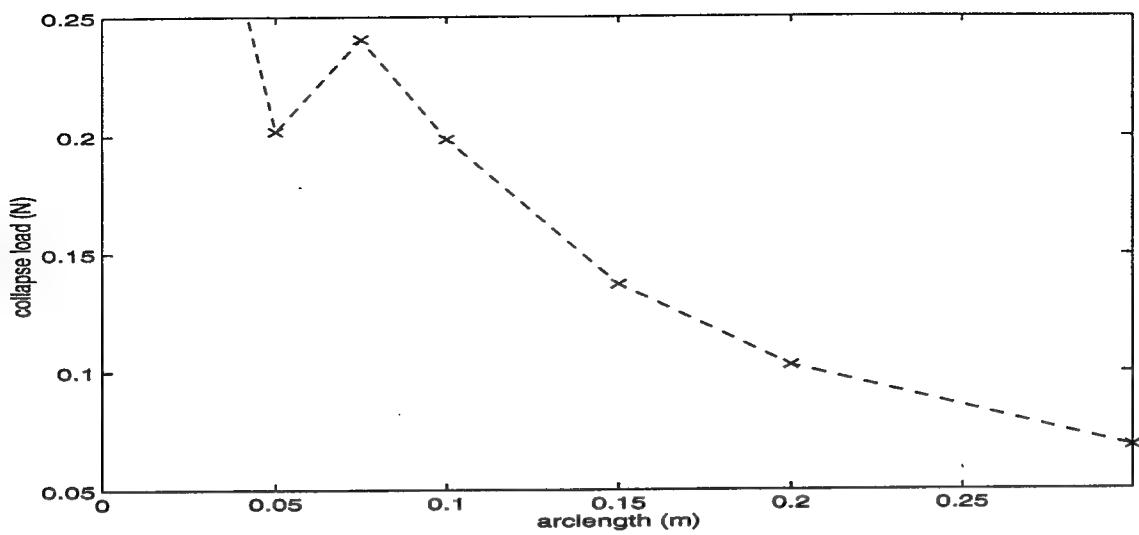
Figure B-3 Critical Load Comparison for $h/r = 0.005$
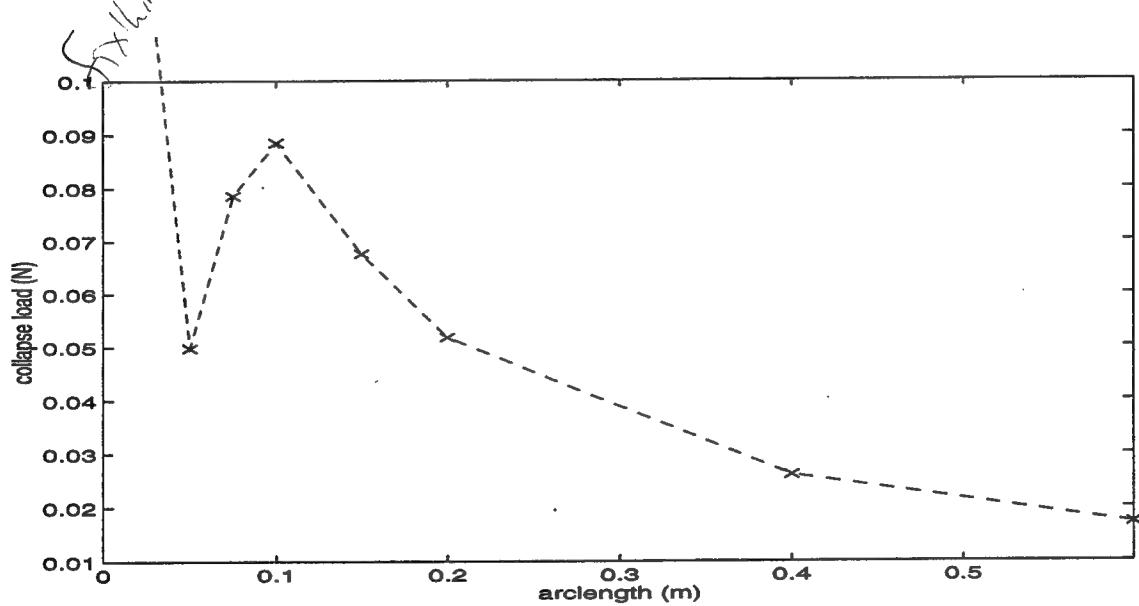
Figure B-4 Critical Load Comparison for $h/r = 0.0025$

Appendix C: Critical Load and Critical Displacement Comparisons for

Constant Arc Length and Variable Thickness Ratio
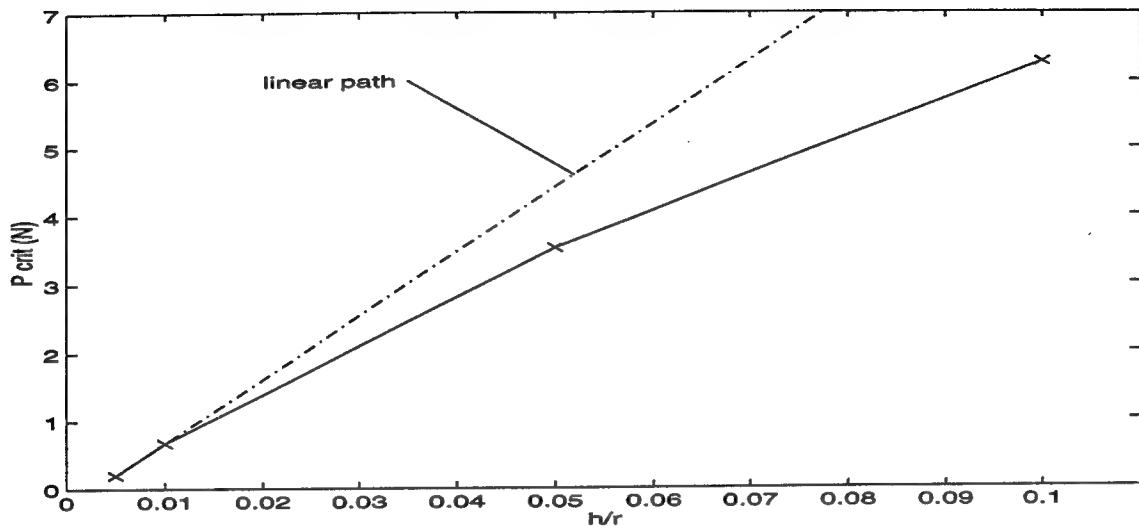
Figure C-1 P_c vs. h/r for $S = 0.05$m
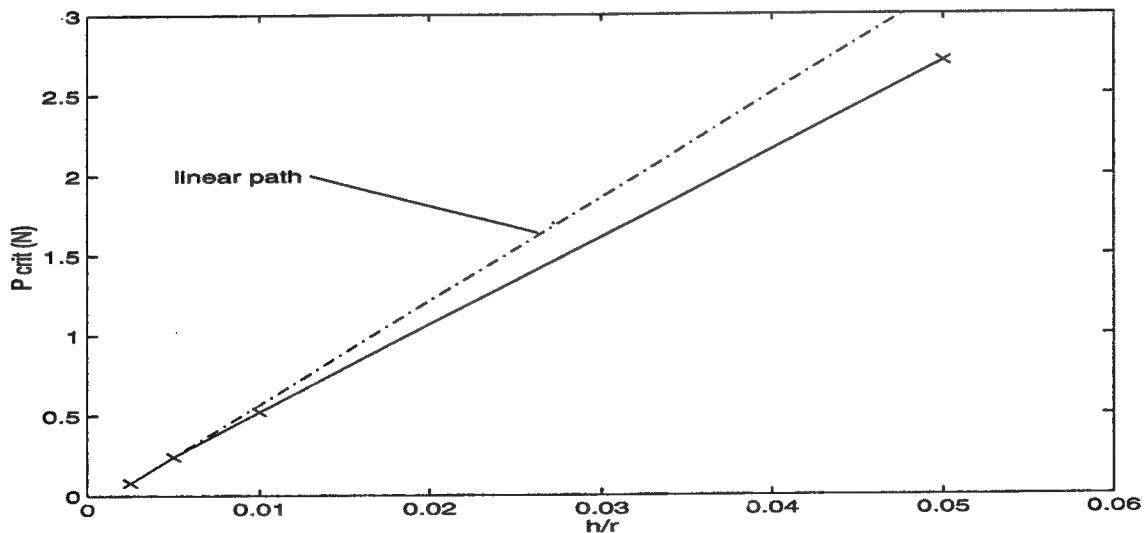
Figure C-2 P_c vs. h/r for $S = 0.075$m
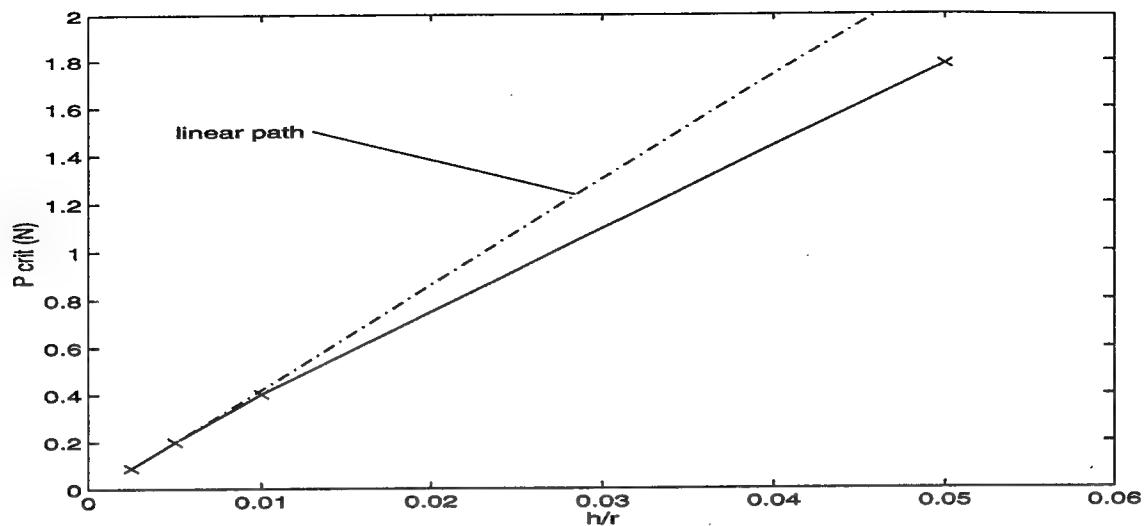
Figure C-3 P_c vs. h/r for $S = 0.1m$
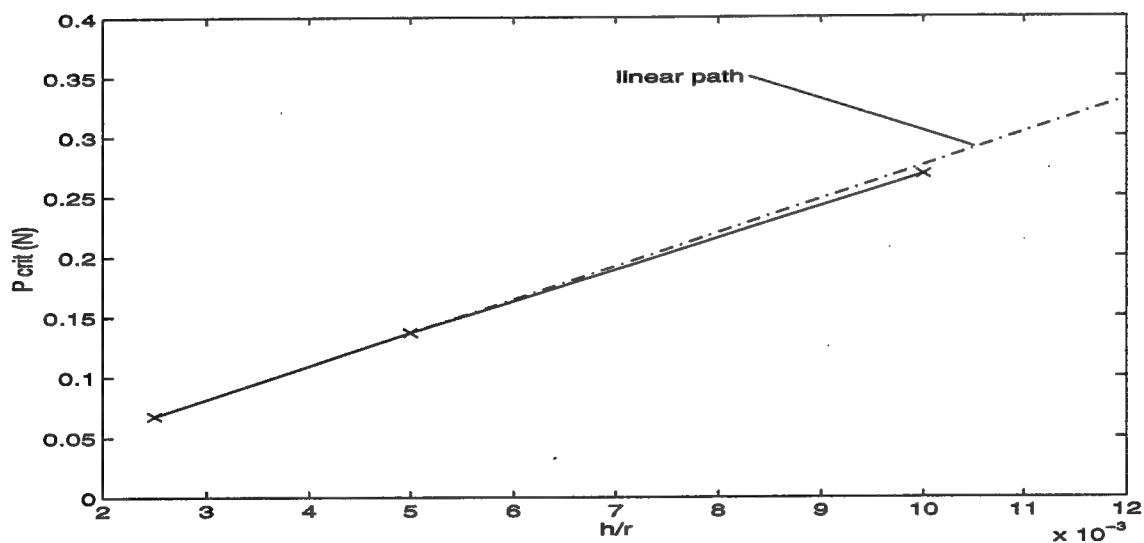
Figure C-4 P_c vs. h/r for $S = 0.15m$
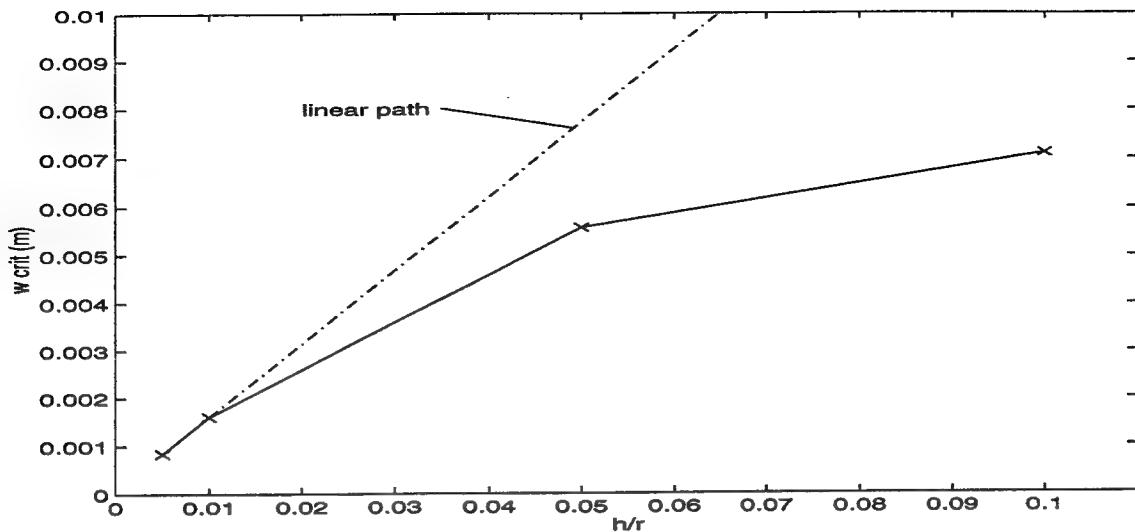
Figure C-5 w_c vs. h/r for $S = 0.05$m
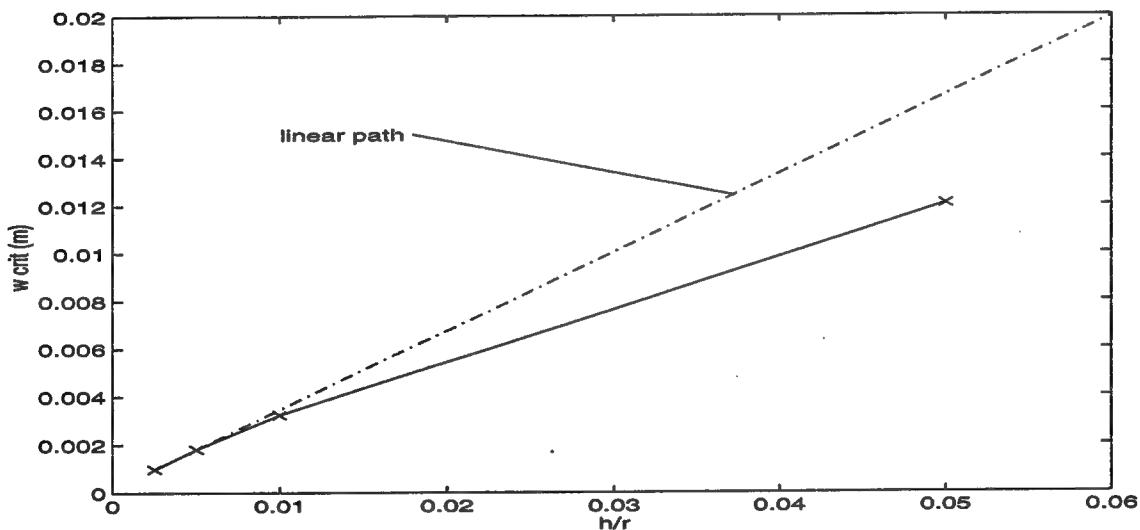
Figure C-6 w_c vs. h/r for $S = 0.075$m
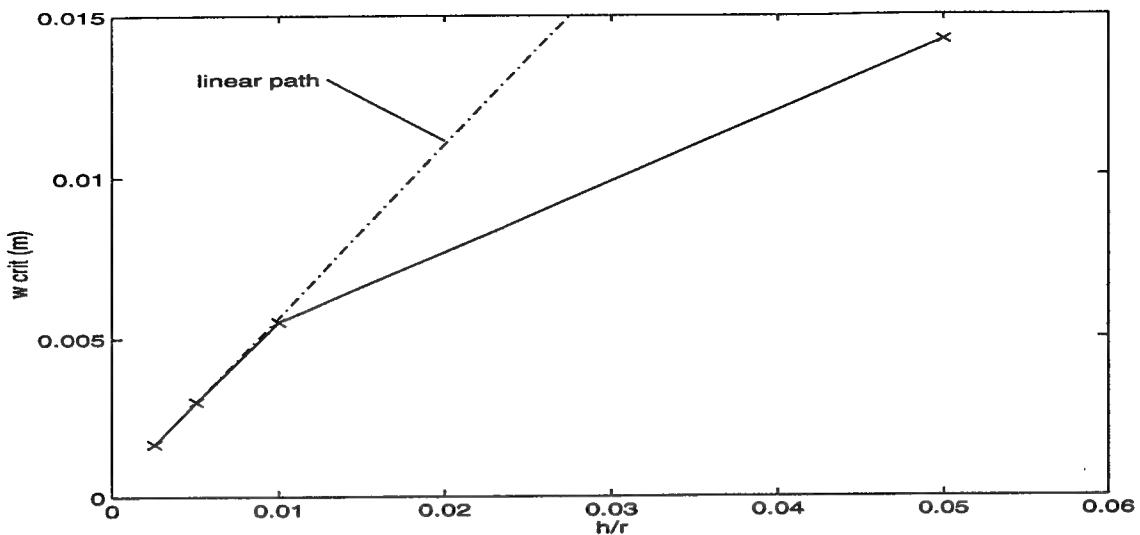
Figure C-7 w_c vs. h/r for $S = 0.1\text{m}$
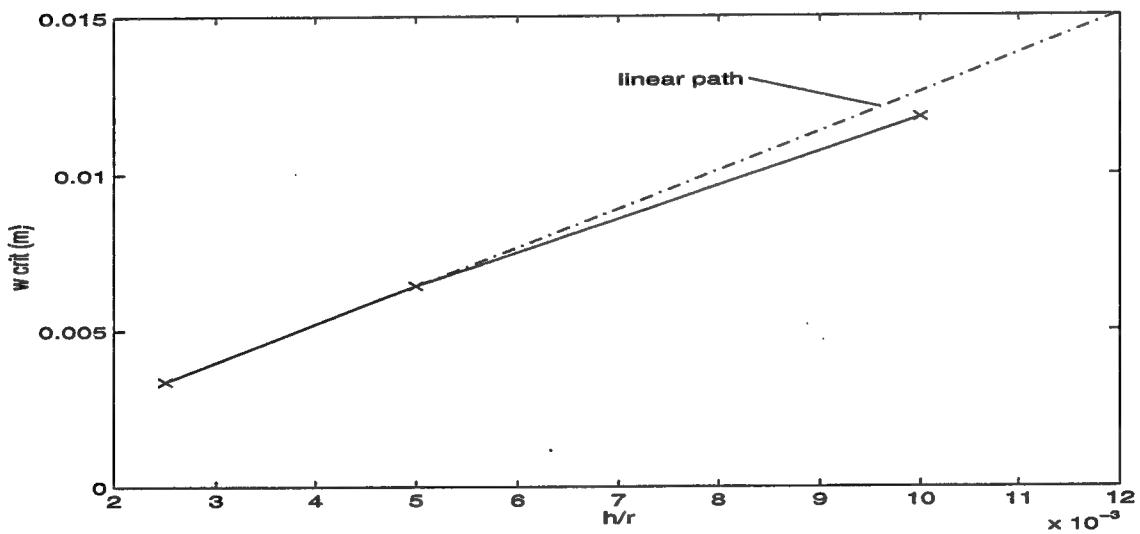
Figure C-8 w_c vs. h/r for $S = 0.15\text{m}$

Appendix D: FORTAN Program Description

D.1 Overview

The following program was initially developed by Creaghan and subsequently modified by Miller (3; 13). The attached listing does not include all of the subroutines, but only the ones modified for the current work. The majority of the modifications were done to incorporate the shape function changes presented in Chapter II. Some minor changes were also incorporated to facilitate multiple simultaneous simulations.

D.2 Data Input Format

This is a line by line description of the user constructed, problem specific data input file required by the program. Each section represents a line and each variable on a line is separated by a comma. Variables are identified by italics.

1.   *title:* problem title text string

2.   *linear, isotro, isarch, ishape:*

    a.   linear: 0 for nonlinear, 1 for linear

    b.   *isotro:* 0 for composite, 1 for isotropic

    c.   *isarch:* 0 for straight beam, 1 for circular arch

    d.   *ishape:* 0 if half a symmetric arch is modeled, 1 for a full arch model

3.   *inctyp, nic, imax, kupdate, tol:*

    a.   *inctyp:* 1 for displacement control, 2 for Riks method

    b.   *ninc:* number of increments (load or displacement) desired

    c.   *imax:* maximum number of iterations per increment

d. *kupdate*: not used, fill with 0 or 1

e. *tol*: convergence tolerance for an increment

4. *pincr, eiter, ttpi*: include only for Riks method nonlinear solutions

a. *pincr*: initial load parameter, λ_0

b. *eiter*: estimated number of iterations per increment, N_s not an integer

c. *ttpi*: maximum load change for an iteration (λ_{max})

5   *table(ninc)*: include only for nonlinear displacement control; this is a list of *ninc* numbers used to attain the incremental global displacement

6. *nelem*: number of elements

7. *delem(nelem)*: list of element lengths; 1 for each element

8. *nbndry*: number of nodes with specified boundary condition (BC)

9. *nbound(nbndry,6)*($v, \psi, \psi_{,2}, w, w_{,2}$): one line for each node with a specified BC the first number is the node number; the next five numbers describe whether or not the dof for the specified node are free or specified in the order indicated; 1=prescribed, 0=free.

10. *vbond(ii)*: prescribed displacements for the fixed dof indicated on the previous line

11. *ldtyp, distld*: not used; fill with 0,0.0

12. *nconc*: number of concentrated loads or moments; Riks needs at least 1.

13. *inconc(nconc)*: use only if *nconc*>0; dof numbers for concentrated loads. middle node dof counts here (11 dof per element) in the following order:

$$v^1, \Psi^1, \Psi^1_{,2}, w^1, w^1_{,2}, v^3, v^2, \Psi^2, \Psi^2_{,2}, w^2, w^2_{,2}$$

14. *vconc*: use only if *nconc*>0; values of loads specified on line 13

15. *ey, nu, ht, width*: include for isotropic material

a. *ey*: Young's modulus

b. *nu*: Poisson's ratio

c. *ht*: thickness

d. *width*: beam or arch width

16. *e1, e2, g12, nu12, g13, g23, width*: include for composite

   materials

   a. *e1*: Young's modulus along fibers

   b. *e2*; Young's modulus transverse to fibers

   c. *g12*: shear modulus g_{12}

   d. *nu12*: Poisson's ratio v_{12}

   e. *g13*: shear modulus g_{13}

   f. *g23*: shear modulus g_{23}

   g. *width*: beam or arch width

17. *nplies, pthick*: include for composite materials

   a. *nplies*: number of plies

   b. *pthick*: individual ply thickness; same for all

18. *theta(nplies)*: include for composites; list of ply orientation

   angles in degrees

19. *rad*: include for curved arch; arch radius of curvature

20. *nforc*: number of nodal resultant forces to be calculated

21. *iforce(nforce)*: include only for *nforc>0*; dof numbers for force

   calculations, include middle node in calculation

22. *nstres*: not used; fill with 0

D.3 Program Listing

```
      program beam
c
c     See bottom of file for variable and subroutine listing.
c        This version is n1 n2 updated to include tangent function and
c        hermitian shape functions for the rotation d.o.f.
c        remember -e compile option (will give 132 character lines)
c        this version is also prompts for plot and shape file names
      implicit double precision (a-h,o-z)
      character*64 gname,fname,pname,sname
c
c
      common/chac/gname,fname
c
      common/elas/ae,de,fe,he,ej,el,re,te,as,ds,fs
      common/input/tol,table(250),delem(250),vbound(2500),distld,
     .  vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
     .  rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
     .  nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
     .  nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
     .  theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
c
      common/stf/stif(11,11),elp(11),eln(11,11),eld(11)
c
      common/proc/gstif(2500,11),gn(2500,11),gf(2500),gd(2500),vperm(250
0),
     .            vpres(2500)
c
      call rinput(pname,sname)
      call elast
      if(inctyp.eq.1)call proces(pname,sname)
      if(inctyp.eq.2)call rikspr(pname,sname)
c********************************************************************
c
c
c     VARIABLES FOR BSHELL
c
c     fname input file
c     gname output file
c     ae,de,        elasticity terms
c     fe,he,        elasticity terms
c     ej,el,        elasticity terms
c     re,te,        elasticity terms
c     as,ds,        elasticity terms
c     fs    elasticity term
c     ey    Young's modulus for isotropic case
c     enu   Poisson's ratio for isotropic case
c     ht    thickness of beam for isotropic case
c     e1,e2,        laminate material properties
c     g12,enu12,    "
c     enu21,g13,    "
c     g23           "
c     pthick        laminate ply thickness
c     nplies        number of plies in laminate
c     theta(20)     ply orientation angles
c     tol   convergence tolerance, percent
c     table(250)    displacement increment multiplicative
c                   factors
c     delem(250)    element lengths
c     vbound(2500)      values of prescribed displacement boundary
```

```
c                   conditions
c       distld        distributed load intensity
c       vconc(2500) concentrated load values
c       rad    arch radius of curvature
c       linear        =1 for linear analysis, =0 for nonlinear
c       isotro        =1 for isotropic, =0 for laminate
c       isarch        =1 for arch, =0 for straight beam
c       ishape        =1 to print x,y coordinates for each node at each
increment
c               when a full arch is represented output to file pname
c       inclod        =1 to increment load(NA), =0 increment displacement
c       ninc   total number of displacement increments
c       imax   maximum number of iterations per increment
c       nelem  total number of elements in model
c       nbndry        number of nodes with specified boundary conditions
c       nbound(250,6)     array of node numbers followed by 1's for
c               fixed b.c.'s, zeros for unfixed
c       ldtyp =1 for distributed load, =0 no distributed load
c       nconc total number of concentrated loads input
c       iconc(2500) DOF's for specified loads
c       nforc number of forces(including moments)to be solved for
c       iforc(2500) DOF's at which to calculate forces
c       nstres        number of elements for stress calculation
c       istres(250) element #'s for stress calculation
c       ibndry(2500)       DOF numbers for b.c.'s
c       idload(250) elements with distributed load
c       coord(251)  coordinate of the nodes
c       width beam or arch width
c       nnod   number of nodes
c********************************************************************************
******
c
c       SUBROUTINES FOR BSHELL
c
c       rinput        reads in and echos input data
c       elast computes elasticity terms
c       proces        drives the solution algorithm for displacement control
c       rikspr(pname,sname)  drives the solution algorithm for Riks method
c       stiff manages stiffness matrix computations
c       shape computes shape function array dsf
c       beamk computes constant stiffness array bmk
c       beamn1        computes linear stiffness array bmn1
c       beamn2        computes quadratic stiffness array bmn2
c       bndy  applies displacement boundary conditions
c       solve solves simultaneous equations in banded array format
c       converge      checks for convergence
c       postpr        computes nodal loads and sends to output file
c
        end
c
c
c
        subroutine rinput(pname,sname)
c
        character*64 fname,gname
        character*4 title
        dimension title(20)
        implicit double precision (a-h,o-z)
c
        common/chac/gname,fname
```

```fortran
      common/input/tol,table(250),delem(250),vbound(2500),distld,
   .    vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
   .    rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
   .    nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
   .    nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
   .    theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
c
      write(*,1000)
      read(*,1005)fname
      write(*,1010)
      read(*,1005)gname
      write(*,1011)
      read(*,1005)pname
      write(*,1012)
      read(*,1005)sname
      open(5,file=fname)
      open(6,file=gname,status='new')
      read(5,1015)title
      read(5,*)linear,isotro,isarch,ishape
      read(5,*)inctyp,ninc,imax,kupdte,tol
      if(linear.eq.0.and.inctyp.eq.2)read(5,*)pincr,eiter,ttpi
      if(linear.eq.0.and.inctyp.eq.1)read(5,*)(table(i),i=1,ninc)
      read(5,*)nelem
      read(5,*)(delem(i),i=1,nelem)
c
c     calculate nodal coordinates
c
      nnod=nelem+1
      coord(1)=0.0
      do 5 ii=2,nnod
    5 coord(ii)=coord(ii-1)+delem(ii-1)
      read(5,*)nbndry
      do 10 i=1,nbndry
   10   read(5,*)(nbound(i,j),j=1,6)
      ifdof=0
c
c     ifdof=counter for enumber of fixed dof's
c
      do 20 i=1,nbndry
      do 20 j=2,6
      if(nbound(i,j).eq.0)goto 20
      ifdof=ifdof+1
      ibndry(ifdof)=(nbound(i,1)-1)*6 + (j-1)
   20   continue
      read(5,*)(vbound(i),i=1,ifdof)
      read(5,*)ldtyp,distld
      if(ldtyp.eq.1)read(5,*)ndload
      if(ldtyp.eq.1)read(5,*)(idload(i),i=1,ndload)
      read(5,*)nconc
      if(nconc.ne.0)read(5,*)(iconc(i),i=1,nconc)
      if(nconc.ne.0)read(5,*)(vconc(i),i=1,nconc)
      if(isotro.eq.1)read(5,*)ey,enu,ht,width
      if(isotro.eq.0)read(5,*)e1,e2,g12,enu12,g13,g23,width
      if(isotro.eq.0)read(5,*)nplies,pthick
      if(isotro.eq.0)read(5,*)(theta(i),i=1,nplies)
      if(isarch.eq.1)read(5,*)rad
      read(5,*)nforc
      if(nforc.ne.0)read(5,*)(iforc(i),i=1,nforc)
      read(5,*)nstres
      if(nstres.ne.0)read(5,*)(istres(i),i=1,nstres)
```

```
c
c      Echo the input to the output file
c
       write(6,1015)title
       if(isarch.eq.1)write(6,1020)
       if(isarch.eq.0)write(6,1025)
       if(linear.eq.1)write(6,1030)
       if(linear.eq.0)write(6,1035)
       if(isotro.eq.1)write(6,1040)
       if(isotro.eq.0)write(6,1045)
       if(ishape.eq.1)write(6,1050)
       if(inctyp.eq.1)write(6,1060)
       if(inctyp.eq.2)write(6,1055)
       write(6,1065)ninc
       write(6,1070)imax
       write(6,1075)tol
       if(inctyp.eq.2)write(6,1076)pincr,eiter,ttpi
       if(inctyp.eq.1)write(6,1078)
       if(inctyp.eq.1)write(6,1080)(table(i),i=1,ninc)
       write(6,1085)nelem
       write(6,1090)
       write(6,1095)(coord(i),i=1,nnod)
       write(6,1100)
       write(6,1105)
       do 30 i=1,nbndry
   30  write(6,1110)(nbound(i,j),j=1,6)
       write(6,1115)ifdof
       write(6,1120)(ibndry(i),i=1,ifdof)
       write(6,1095)(vbound(i),i=1,ifdof)
       if(ldtyp.eq.1)write(6,1125)distld
       if(ldtyp.eq.1)write(6,1130)(idload(i),i=1,ndload)
       if(nconc.ne.0)write(6,1135)
       if(nconc.ne.0)write(6,1120)(iconc(i),i=1,nconc)
       if(nconc.ne.0)write(6,1095)(vconc(i),i=1,nconc)
       if(isotro.eq.1)write(6,1140)ey,enu,ht,width
       if(isotro.eq.0)write(6,1145)e1,e2,g12,enu12,g13,g23,width
       if(isotro.eq.0)write(6,1150)nplies,pthick
       if(isotro.eq.0)write(6,1155)(theta(i),i=1,nplies)
       if(isarch.eq.1)write(6,1160)rad
       write(6,1165)(iforc(i),i=1,nforc)
       write(6,1170)(istres(i),i=1,nstres)
       close(5)
c      close(6)
c
c
c      F  O  R  M  A  T  S
c
 1000 format('Enter your input file name.')
 1005 format(A)
 1010 format('Enter your output file name.')
 1011 format('Enter your plot file name.')
 1012 format('Enter your shape file name.')
 1015 format(20a4)
 1020 format(/,1x,'Element type: arch')
 1025 format(/,1x,'Element type: straight beam')
 1030    format(/,1x,'Analysis type: linear')
 1035 format(/,1x,'Analysis type: nonlinear')
 1040 format(/,1x,'Material type: isotropic')
 1045 format(/,1x,'Material type: laminate')
 1050 format(/,1x,'Printout of nodal x,y coordinates requested')
```

```fortran
1055 format(/,1x,'Riks method specified')
1060 format(/,1x,'Displacement control method specified')
1065 format(/,1x,'Increments specified:',2x,i3)
1070 format(/,1x,'Maximum iterations specified:',2x,i3)
1075 format(/,1x,'Percent convergence tolerance:',2x,d12.5)
1076 format(/,1x,'pincr=',2x,d12.5,2x,'eiter=',2x,d12.5,2x,
     .       'ttpi=',2x,d12.5)
1078 format(/,1x,'Displacement Increment Table')
1080 format(8(2x,d12.5))
1085 format(/,1x,'Number of elements:',2x,i3)
1090 format(/,1x,'Nodal Coordinates:')
1095 format(8(2x,d12.5))
1100 format(/,1x,'DISPLACEMENT BOUNDARY CONDITIONS, 1=PRESCRIBED,
     X0=FREE')
1105 format(/,4X,'NODE    V   PSI-S  PSI-SS   W    W-S  ')
1110 format(4x,i4,3x,5(i3,3x))
1115 format(/,1x,'NUMBER OF PRESCRIBED DISPLACEMENTS:',
     .    i5,/,1x,'SPECIFIED DISPLACEMENT DOF AND THIER
     .    VALUES FOLLOW:')
1120 format(16i5)

1125 format(/,1x,'Distributed Load Intensity:',2x,d12.5)
1130 format(/,1x,'Elements with distributed load:',/,1x,16i5)
1135 format(/,1x,'DOF and specified concentrated loadsfollow:')
1140 format(/,1x,'Isotropic material properties ey, enu, ht, width:'
     .       ,/,1x,4d12.5)
1145 format(/,1x,'Composite material properties e1, e2, g12, enu12,
     .    g13,g23, width:',/,1x,7d12.5)
1150 format(/,1x,'Number of plies:',2x,i3,2x,'Ply thickness:',2x,
     .    d12.5)
1155 format(/,1x,'Ply orientation angles:',/,1x,8(2x,d12.5))
1160 format(/,1x,'Radius of curvature:',2x,d12.5)
1165 format(/,1x,'DOFs for equivalent load calculation:',/,
     .    1x,16i5)
1170 format(/,1x,'Elements for stress calculation:',/,1x,16i5)
1175 format(/,1x,i5)
     return
     end
c
c
c
     subroutine elast
c
     implicit double precision (a-h,o-z)
c
     dimension qbar(3,3),rtheta(20)
c
c
     character*64 gname,fname
     common/chac/gname,fname
     common/elas/ae,de,fe,he,ej,el,re,te,as,ds,fs
     common/input/tol,table(250),delem(250),vbound(2500),distld,
     .   vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
     .   rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
     .   nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
     .   nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
     .   theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
c
c
c     Isotropic case
```

```
c
c       write(6,1000)e1,e2,g12,enu12,enu21,g13,g23,pthick
        if(isotro.eq.0)goto 100
        gs=ey/(2*(1+enu))
                denom=1.-enu**2
                q11=ey/denom
        q12=enu*ey/denom
        q22=q11
        q2hat=q22-(q12**2/q11)
        qs4=gs
        ae=q2hat*ht
                de=q2hat*ht**3/(3*2.**2)
                fe=q2hat*ht**5/(5*2.**4)
                he=q2hat*ht**7/(7*2.**6)
                ej=q2hat*ht**9/(9*2.**8)
                el=q2hat*ht**11/(11*2.**10)
                re=q2hat*ht**13/(13*2.**12)
        te=q2hat*ht**15/(15*2.**14)
        as=qs4*ht
        ds=qs4*ht**3/(3*2.**2)
        fs=qs4*ht**5/(5*2.**4)
        goto 200
c
c       Laminate case
c
 100    ht=pthick*nplies
                enu21=e2*enu12/e1
                denom=1.-enu12*enu21
                q11=e1/denom
                q12=enu12*e2/denom
                q22=e2/denom
c
c***************************************************
c       calculate the elasticity matrices           *
c                                                     *
c       remem that the z axis points down,          *
c       however, the first ply is the top ply, ie,  *
c       the ply with the most negative z !!!        *
c***************************************************
c
c       initialize elasticity terms
c
        ae=0.
                de=0.
                fe=0.
                he=0.
                ej=0.
                el=0.
                re=0.
        te=0.
        as=0.
        ds=0.
        fs=0.
                do 45 ii=1,nplies
 45             rtheta(ii)=theta(ii)*3.14159265/180.
                do 50 kk=1,nplies

        qbar(1,1)=q11*(cos(rtheta(kk))**4)+2*q12*(sin(rtheta(kk))**2)*
     .      (cos(rtheta(kk))**2)+q22*(sin(rtheta(kk))**4)
```

```
      qbar(1,2)=(q11+q22)*(sin(rtheta(kk))**2)*(cos(rtheta(kk))**2)+
     .          q12*(sin(rtheta(kk))**4+cos(rtheta(kk))**4)

      qbar(2,2)=q11*(sin(rtheta(kk))**4)+2*q12*(sin(rtheta(kk))**2)*
     .          (cos(rtheta(kk))**2)+q22*cos(rtheta(kk))**4
         qs4=g13*dcos(rtheta(kk))**2+g23*dsin(rtheta(kk))**2
      q2hat=qbar(2,2)-(qbar(1,2)**2/qbar(1,1))
      zl=(kk*1. - nplies*.5)*pthick
         zu=zl-pthick
      ae=ae + q2hat*pthick
      de=de + q2hat*(zl**3-zu**3)/3.
         fe=fe + q2hat*(zl**5-zu**5)/5.
         he=he + q2hat*(zl**7-zu**7)/7.
         ej=ej + q2hat*(zl**9-zu**9)/9.
       el=el + q2hat*(zl**11-zu**11)/11.
         re=re + q2hat*(zl**13-zu**13)/13.
         te=te + q2hat*(zl**15-zu**15)/15.
      as=as+qs4*pthick
         ds=ds+qs4*(zl**3-zu**3)/3.
      fs=fs+qs4*(zl**5-zu**5)/5.
  50  continue
c 200 open(6,fiel=gname,status='old')
  200 write(6,1000)ae,de,fe,he,ej,el,re,te,as,ds,fs
c     close(6)
 1000 format(/,1x,'Elasticity terms:',/,1x,8(2x,d12.5))
      return
      end
c
c
c

      subroutine proces(pname,sname)
c
      implicit double precision (a-h,o-z)
c
      character*64 gname,fname
c
c
      common/chac/gname,fname
c
      common/elas/ae,de,fe,he,ej,el,re,te,as,ds,fs
c
      common/input/tol,table(250),delem(250),vbound(2500),distld,
     .  vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
     .  rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
     .  nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
     .  nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
     .  theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
      common/stf/stif(11,11),elp(11),eln(11,11),eld(11)
c
      common/proc/gstif(2500,11),gn(2500,11),gf(2500),gd(2500),vperm(250
0),
     .          vpres(2500)
c
      ndof=nnod*5+nelem
      ncount=1
      icount=1
      do 1 ii=1,ndof
  1   gd(ii)=0.0d0
      do 2 ii=1,nbndry*6
```

```fortran
         vpres(ii)=0.0d0
   2     vperm(ii)=vbound(ii)
c
c        start new increment or iteration/
c        zero out global stiffness matrices and global force
c        vector
c
   3     do 5 ii=1,ndof
         gf(ii)=0.0d0
         do 5 jj=1,11
         gstif(ii,jj)=0.0d0
   5     gn(ii,jj)=0.0d0
         kcall=0
c
c        increment prescribed displacement for displacement control
c
         if(linear.eq.1)goto 9
         if(icount.ne.1)goto 9
         do 7 ii=1,nbndry*6
         if(ncount.eq.1)vbound(ii)=vperm(ii)*table(1)
   7     if(ncount.gt.1)vbound(ii)=vperm(ii)*(table(ncount)-
     .           table(ncount-1))
c
c        loop over all elements for stiffness and forces
c
   9     do 30 ielem=1,nelem
         do 10 ii=1,11
  10     eld(ii)=gd(ii+(ielem-1)*6)
c
         kcall=kcall+1
         call stiff(ielem,icount,ncount,kcall)
c
c        Assemble global stiffness array, gstif, global equilibrium
c        stiffness, gn, in banded form. Half-bandwidth=11.  Also
c        assemble global force vector, gf.
c
         nr=(ielem-1)*6 + 1
         do 30 jj=0,10
         gf(nr+jj)=gf(nr+jj)+elp(jj+1)
         do 30 kk=1,11-jj
         gstif(nr+jj,kk)=gstif(nr+jj,kk)+stif(jj+1,kk+jj)
         if(linear.eq.1)goto 30
         if(icount.eq.1 .and. ncount.eq.1)goto 30
         gn(nr+jj,kk)=gn(nr+jj,kk)+eln(jj+1,kk+jj)
  30     continue
c
c        impose force boundary conditions
c        at this point, gf=R
c
         if(nconc.eq.0)goto 45
         do 40 ii=1,nconc
         nb=iconc(ii)
  40     gf(nb)=gf(nb)+vconc(ii)
  45     continue
c
c        calculate the residual force vector for nonlinear
c        analysis. -[gn]*{gd}+R=-[k+n1/2+n2/3]*{q}+R=gf
c
             if(icount.eq.1)goto 65
             do 60 ii=1,ndof
```

```
                  add=0.
                  do 50 kk=1,ii-1
                  if(ii-kk+1 .gt. 11)goto 50
                  add=add+gn(kk,ii-kk+1)*gd(kk)
        50        continue
                  res=0.
                  do 55 jj=1,11
                  if(jj+ii-1 .gt. ndof)goto 55
                  res=res + gn(ii,jj)*gd(jj+ii-1)
        55        continue
c
c       add to existing gf which already contains R
c
        gf(ii)=gf(ii)-res-add
      60 continue
      65 continue
c
c       impose displacement boundary conditions
c
        if(icount.eq.1)call bndy(ndof,gstif,gf,nbndry,ibndry,vbound)
        if(icount.gt.1)call bndy(ndof,gstif,gf,nbndry,ibndry,vpres)
c
c       solve system of equations, result in gf
c
        call solve(ndof,gstif,gf,0,detm,detm1)
c
c       update total displacement vector gd
c
        do 70 ii=1,ndof
      70 gd(ii)=gd(ii)+gf(ii)
        if(linear.eq.1)goto 80
        call converge(ndof,ncon,icount,tol,imax)
c
c       if no convergence (ncon=0) start next iteration
c
        if(ncon.eq.0)goto 3
        80        continue
        if(ncon.eq.1 .and. ncount.le.ninc)then
          call postpr(icount,ncount,kcall,ndof,pname,sname)
          if(ncount.eq.ninc)stop
          ncount=ncount+1
          icount=1
          goto 3
        endif
        return
        end
c
        subroutine bndy(ndof,s,sl,ndum,idum,vdum)
c
c       ....................................................................
c       subroutine used to impose boundary conditions on banded equations
c       ....................................................................
c
        implicit double precision (a-h,o-z)
        dimension s(2500,11),sl(2500)
        dimension idum(ndum*6),vdum(ndum*6)
        do 300 nb = 1, ndum*6
        ie = idum(nb)
        sval = vdum(nb)
        it=10
```

```
      i=ie-11
      do 100 ii=1,it
      i=i+1
      if (i .lt. 1)  go to 100
      j=ie-i+1
      sl(i)=sl(i)-s(i,j)*sval
      s(i,j)=0.0
  100 continue
      s(ie,1)=1.0
      sl(ie)=sval
      i=ie
      do 200 ii=2,11
      i=i+1
      if (i .gt. ndof)  go to 200
      sl(i)=sl(i)-s(ie,ii)*sval
      s(ie,ii)=0.0
  200 continue
  300 continue
      return
      end
c
c
c
c
      subroutine converge(ndof,ncon,icount,tol,imax)
c........................................................
c checks for convergnece using global displacement criterion
c........................................................
      implicit double precision (a-h,o-z)
      common/proc/gstif(2500,11),gn(2500,11),gf(2500),gd(2500),vperm(250
0),
      .            vpres(2500)
c
      rcurr=0.
      do 10 m=1,ndof
  10  rcurr=rcurr + gd(m)*gd(m)
      if(icount.eq.1)rinit=rcurr
      if(icount.eq.1)ncon=0
      if(icount.eq.1)goto 20
c new criteria
      ratio=100. * abs(sqrt(rcurr)-sqrt(pvalue))/sqrt(rinit)
      if(ratio.le.tol)ncon=1
  20  pvalue=rcurr
      write(*,100)ncon,ratio,rinit,rcurr
  100 format(1x,'ncon= ',i3,3x,'ratio= ',d14.6,' rinit= ',d14.6,
     x           ' rcurr= ',d14.6)
      if(icount.eq.imax)write(6,200)
      if(icount.eq.imax)stop
  200 format(1x,'icount equals imax')
      if(ncon.eq.0)icount=icount+1
      return
      end
c
c
c
      subroutine rikspr(pname,sname)
c
      implicit double precision (a-h,o-z)
c
      character*64 gname,fname
```

```
c
c
        common/chac/gname,fname
c
        common/elas/ae,de,fe,he,ej,el,re,te,as,ds,fs
c
        common/input/tol,table(250),delem(250),vbound(2500),distld,
     .    vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
     .    rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
     .    nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
     .    nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
     .    theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
        common/stf/stif(11,11),elp(11),eln(11,11),eld(11)
c
        common/proc/gstif(2500,11),gn(2500,11),gf(2500),gd(2500),.vperm(250
0),
     .            vpres(2500)
c
        dimension
gld(2500),gld0(2500),gld1(2500),gdis(2500),gsti00(2500,11),
     .            gf0(2500),gd00(2500)
        ndof=nnod*5+nelem
        ncount=1
        icount=1
        iicut=0
        do 1 ii=1,ndof
    1   gd(ii)=0.0d0
        do 2 ii=1,nbndry*6
        vpres(ii)=0.0d0
    2   vperm(ii)=vbound(ii)
c
c       start new increment or iteration/
c       zero out global stiffness matrices and global force
c       vector
c
        tpincr=0.0
        if(ncount.eq.1)goto 2993
c
c       start new increment
c
    3   if(iicut.eq.0)dss=dss*eiter/icount
 2993   icount=1
        do 2992 ii=1,ndof
        gld0(ii)=0.0d0
 2992   gd00(ii)=gd(ii)
c
c       start new iteration
c
    4   do 5 ii=1,ndof
        gf0(ii)=0.0d0
        do 5 jj=1,11
        gstif(ii,jj)=0.0d0
    5   gn(ii,jj)=0.0d0
        kcall=0
c
c       increment prescribed displacement for displacement control
c
c       if(linear.eq.1)goto 9
c       if(icount.ne.1)goto 9
c       do 7 ii=1,nbndry*6
```

```
c        if(ncount.eq.1.and.iicut.eq.0)vbound(ii)=vperm(ii)*table(1)
c    7   if(ncount.gt.1.or.iicut.gt.0)vbound(ii)=vperm(ii)*(table(ncount)-
c      .        table(ncount-1))
c
c        loop over all elements for stiffness and forces
c
   9    do 30 ielem=1,nelem
        do 10 ii=1,11
  10    eld(ii)=gd(ii+(ielem-1)*6)
c
        kcall=kcall+1
        call stiff(ielem,icount,ncount,kcall)
c
c        Assemble global stiffness array, gstif, global equilibrium
c        stiffness, gn, in banded form. Half-bandwidth=11.  Also
c        assemble global force vector, gf.
c
        nr=(ielem-1)*6 + 1
        do 30 jj=0,10
        gf0(nr+jj)=gf0(nr+jj)+elp(jj+1)
        do 30 kk=1,11-jj
        gstif(nr+jj,kk)=gstif(nr+jj,kk)+stif(jj+1,kk+jj)
        if(linear.eq.1)goto 30
        if(icount.eq.1 .and. ncount.eq.1 .and.iicut.eq.0)goto 30
        gn(nr+jj,kk)=gn(nr+jj,kk)+eln(jj+1,kk+jj)
  30    continue
c
c        impose force boundary conditions
c        at this point, gf=R
c
        if(nconc.eq.0)goto 45
        do 40 ii=1,nconc
        nb=iconc(ii)
  40    gf0(nb)=gf0(nb)+vconc(ii)
  45    continue
        do 47 ii=1,ndof
  47    gf(ii)=gf0(ii)
        do 48 ii=1,ndof
        do 48 jj=1,11
  48    gsti00(ii,jj)=gstif(ii,jj)
        call bndy(ndof,gstif,gf,nbndry,ibndry,vbound)
c
        call solve(ndof,gstif,gf,0,detm,detm1)
        dss0=0.0
        do 49 ii=1,ndof
        gdis(ii)=gf(ii)
  49    dss0=dss0+gf(ii)*gf(ii)
        if(icount.ne.1) go to 144
            detm1=detm2
            detm2=detm
            if(ncount.eq.1.and.detm.lt.0.0 .and.iicut.eq.0) pincr=-pincr
            if(ncount.eq.1.and.iicut.eq.0) dss=pincr*dsqrt(dss0)
            if(ncount.ne.1.or.iicut.gt.0) pincr=
dss/dsqrt(dss0)*detm*detm1
            .                            *pincr1/dabs(pincr1)
c
c        attempt at offloading at bifurcation points
c
c        if(iicut.eq.1)pincr=-pincr
c
```

```fortran
             pincr1=pincr

             prs=0.0
             do 142 ii=1,ndof
  142          prs=prs+gf0(ii)*gld(ii)
       stifpa=pincr*prs
       do 143 ii=1,ndof
  143 gld(ii)=pincr*gdis(ii)
  144 continue
c
c     calculate the residual force vector for nonlinear
c     analysis.  -[gn]*{gd}+R=-[k+n1/2+n2/3]*{q}+R=gf
c

       if(icount.eq.1)goto 69
             do 60 ii=1,ndof
             add=0.
             do 50 kk=1,ii-1
             if(ii-kk+1 .gt. 11)goto 50
             add=add+gn(kk,ii-kk+1)*gd(kk)
   50        continue
             res=0.
       do 55 jj=1,11
       if(jj+ii-1 .gt. ndof)goto 55
             res=res + gn(ii,jj)*gd(jj+ii-1)
   55        continue
c
c     add to existing gf which already contains R
c
       gf(ii)=gf0(ii)*(pincr+tpincr)-res-add
   60 continue
   65 continue
c
c     impose displacement boundary conditions
c
       call bndy(ndof,gsti00,gf,nbndry,ibndry,vbound)
c      if(icount.gt.1)call bndy(ndof,gstif,gf,nbndry,ibndry,vpres)
c
c     solve system of equations, result in gf
c
       call solve(ndof,gstif,gf,1,detm,detml)
c
c     through line 69 copied from Tsai's program
c
       a1=dss0
       a2=0.0
       a3=0.0
       do 147 ii=1,ndof
       a2=a2+(gld(ii)+gf(ii))*gdis(ii)
  147 a3=a3+gf(ii)*(2.0*gld(ii)+gf(ii))
       d12=a2*a2-a1*a3
c      write(6,*) d12,a1,a2,a3
c
c
       if(d12.lt.0.0)then
c
c     deal with complex roots  by cutting the search
c     radius (dss) in half
c
         do 2991 ii=1,ndof
```

```
2991 gd(ii)=gd00(ii)
     iicut=iicut + 1
     if(iicut.gt.10)then
          write(6,3000)
          stop
          endif
     dss=dss/2.0
     goto 3
     endif
     iicut=0
     dpinc1=(-a2+dsqrt(d12))/a1
     dpinc2=(-a2-dsqrt(d12))/a1
     theta1=0.0
     theta2=0.0
     do 148 ii=1,ndof
     gld0(ii)=gld(ii)
     gld(ii)=gld(ii)+gf(ii)+dpinc1*gdis(ii)
     gld1(ii)=gld0(ii)+gf(ii)+dpinc2*gdis(ii)
     theta1=theta1+gld0(ii)*gld(ii)
     theta2=theta2+gld0(ii)*gld1(ii)
 148 continue
c     write(6,*) theta1,theta2
     thet12=theta1*theta2
     if(thet12.gt.0.0) go to 149
     dpincr=dpinc1
     if(theta2.gt.0.0) call chsign(gld,gld1,dpincr,dpinc2,ndof)
     go to 150
 149 dpib=-a3/(a2*2.0)
     dpin1=dabs(dpib-dpinc1)
     dpin2=dabs(dpib-dpinc2)
     dpincr=dpinc1
     if(dpin2.lt.dpin1) call chsign(gld,gld1,dpincr,dpinc2,ndof)
 150 pincr=pincr+dpincr
  69 continue
c
c     update total displacement vector gd
c
     do 70 ii=1,ndof
  70 gd(ii)=gd(ii)+gld(ii)-gld0(ii)
     if(linear.eq.1)goto 80
     call converge(ndof,ncon,icount,tol,imax)
c
c     if no convergence (ncon=0) start next iteration
c
     if(ncon.eq.0)goto 4
  80      continue
     if(ncon.eq.1 .and. ncount.le.ninc)then
       call postpr(icount,ncount,kcall,ndof,pname,sname)
       if(ncount.eq.ninc)stop
       ncount=ncount+1
       tpincr=tpincr+pincr
       goto 3
     endif
3000     format(1x,'More than 10 consecutive imaginary roots')
3010 format(/,1x,i2)
     return
     end
c
c
c
```

```fortran
      subroutine solve(ndof,band,rhs,ires,detm,detml)
c     .................................................
c     solve a banded symmetric system of equations
c     .................................................
c
      implicit double precision (a-h,o-z)
      dimension band(2500,11),rhs(2500)
      meqns=ndof-1
      if(ires.gt.0)goto 90
      do 500 npiv=1,meqns
c      print*,'npiv= ',npiv
      npivot=npiv+1
      lstsub=npiv+11-1
      if(lstsub.gt.ndof) lstsub=ndof
      do 400 nrow=npivot,lstsub
c     invert rows and columns for row factor
      ncol=nrow-npiv+1
      factor=band(npiv,ncol)/band(npiv,1)
      do 200 ncol=nrow,lstsub
      icol=ncol-nrow+1
      jcol=ncol-npiv+1
  200 band(nrow,icol)=band(nrow,icol)-factor*band(npiv,jcol)
  400 rhs(nrow)=rhs(nrow)-factor*rhs(npiv)
  500 continue
      detm=1.0
      detml=0.0
      do 600 ii=1,ndof
c      write (*,*) 'BAND(',ii,'1)=',band(ii,1)
      detml=detml+dlog10(dabs(band(ii,1)))
  600 detm=detm*band(ii,1)/dabs(band(ii,1))
      go to 101
   90 do 100 npiv=1,meqns
      npivot=npiv+1
      lstsub=npiv+11-1
      if(lstsub.gt.ndof) lstsub=ndof
      do 110 nrow=npivot,lstsub
      ncol=nrow-npiv+1
      factor=band(npiv,ncol)/band(npiv,1)
  110 rhs(nrow)=rhs(nrow)-factor*rhs(npiv)
  100 continue
c     back substitution
  101 do 800 ijk=2,ndof
      npiv=ndof-ijk+2
      rhs(npiv)=rhs(npiv)/band(npiv,1)
      lstsub=npiv-11+1
      if(lstsub.lt.1) lstsub=1
      npivot=npiv-1
      do 700 jki=lstsub,npivot
      nrow=npivot-jki+lstsub
      ncol=npiv-nrow+1
      factor=band(nrow,ncol)
  700 rhs(nrow)=rhs(nrow)-factor*rhs(npiv)
  800 continue
      rhs(1)=rhs(1)/band(1,1)
      return
      end
c
c
c
      subroutine chsign(gld,gld1,dpincr,dpinc2,ndof)
```

```
        implicit double precision (a-h,o-z)
        dimension gld(2500),gld1(2500)
        do 100 i=1,ndof
 100    gld(i)=gld1(i)
        dpincr=dpinc2
        return
        end
c
c
c
        subroutine stiff(ielem,icount,ncount,kcall)
c
        implicit double precision (a-h,o-z)
        character*64 gname,fname
        common/chac/gname,fname
c
        common/input/tol,table(250),delem(250),vbound(2500),distld,
      .   vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
      .   rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
      .   nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
      .   nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
      .   theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
c
        common/elas/ae,de,fe,he,ej,el,re,te,as,ds,fs
c
        common/stf/stif(11,11),elp(11),eln(11,11),eld(11)
c
        common/shp/dsf(7,11)
c
        dimension bmk(7,7),bmn1(7,7),bmn2(7,7),
      .   gauss4(4),wt4(4),gauss7(7),wt7(7),q(7),dsftr(11,7),
      .       pkt(7,7),pkn(7,7),pktd(7,11),pknd(7,11),gauss5(5),wt5(5)
c
        data gauss4/0.8611363115d0,0.3399810435d0,-0.3399810435d0,
      .     -0.8611363115d0/
        data wt4/0.3478548451d0,0.6521451548d0,0.6521451548d0,
      .     0.3478548451d0/
        data gauss5/0.9061798459d0,0.5384693101d0,0.0d0,-0.5384693101d0,
      .     -0.9061798459d0/
        data wt5/0.2369268851d0,0.4786286705d0,0.5688888889d0,
      .     0.4786286705d0,0.2369268851d0/
        data gauss7/0.9491079123d0,0.7415311856d0,0.4058451513d0,
      .     0.0d0,-0.4058451513d0,-0.7415311856d0,-0.9491079123d0/
        data wt7/0.1294849662d0,0.2797053915d0,0.3818300505d0,
      .     0.4179591836d0,0.3818300505d0,0.2797053915d0,0.1294849662d0/
c
c       initialize stiffness arrays and load array
c
        do 10 ii=1,11
        elp(ii)=0.0
        do 10 jj=1,11
        stif(ii,jj)=0.0
   10   eln(ii,jj)=0.0
c
c       set number of gauss points for interpolation
c
        ngp=5
        if(ncount.eq.1 .and. icount.eq.1)ngp=4
        if(linear.eq.1)ngp=4
c
```

```fortran
      ek1=-4./(3.*ht**2)
      if (isarch.eq.1) p1=1./rad
      if(ncount.eq.1 .and. icount.eq.1 .and. kcall.eq.1 .and.
 isarch.eq.1)
     .     call beamk(bmk,ek1,p1)
      if(ncount.eq.1 .and. icount.eq.1 .and. kcall.eq.1 .and.
 isarch.eq.0)
     .     call sbeamk(bmk,ek1)
c
c     loop over gauss points
c
      do 100 ii=1,ngp
      if(ngp.eq.4)eta=gauss4(ii)
      if(ngp.eq.5)eta=gauss5(ii)
      if(ngp.eq.7)eta=gauss7(ii)
      call shape(eta,ielem,aa)
c
c     multiply element displacement vector, eld (this is 'q'
c     in the thesis) by the shape function matrix, dsf, to get
c     the displacement gradient vector(d(s) in thesis, q here)
c
      do 20 kk=1,7
   20 q(kk)=0.0
c
      do 30 jj=1,7
      do 30 kk=1,11
   30 q(jj)=q(jj)+dsf(jj,kk)*eld(kk)
c
c
c     initialize bmn1, bmn2
c
      do 35 kk=1,7
      do 35 jj=1,7
      bmn1(jj,kk)=0.0d0
   35 bmn2(jj,kk)=0.0d0
c
c     skip bmn1 and bmn2 comps first time through
c
      if(icount .eq. 1 .and. ncount .eq. 1)goto 37
c
      if(isarch.eq.1)call beamn1(q,bmn1,ek1,p1)
      if(isarch.eq.1)call beamn2(q,bmn2,ek1,p1)
      if(isarch.eq.0)call sbmn1(q,bmn1,ek1)
      if(isarch.eq.0)call sbmn2(q,bmn2,ek1)
   37    continue
c
c     transpose the shape function matrix
c
      do 40 jj=1,7
      do 40 kk=1,11
   40 dsftr(kk,jj)=dsf(jj,kk)
c
c     create element independent incremental stiffness array,
c     pkt, and element ind. equilibrium stiffness array, pkn
c
      do 50 jj=1,7
      do 50 kk=1,7
      pkt(jj,kk)=bmk(jj,kk)+bmn1(jj,kk)+bmn2(jj,kk)
   50 pkn(jj,kk)=bmk(jj,kk)+bmn1(jj,kk)/2.+bmn2(jj,kk)/3.
c
```

```fortran
c        post-multiply each array by the shape function matrix
c
         do 60 jj=1,7
         do 60 kk=1,11
         pktd(jj,kk)=0.0
         pknd(jj,kk)=0.0
         do 60 ll=1,7
         pktd(jj,kk)=pktd(jj,kk) + aa*pkt(jj,ll)*dsf(ll,kk)
   60 pknd(jj,kk)=pknd(jj,kk) + aa*pkn(jj,ll)*dsf(ll,kk)
c
c        Finally, pre-multiply these new arrays by the transpose
c        of the shape function matrix to get the element incremental
c        stiffness, stif, and element equilibrium stiffness, eln.
c        Also multiply by the weighting factor for this particular
c        gauss point.  Note that these arrays are zeroed outside the
c        loop over the gauss points since they accumulate (integrate)
c        data over all the gauss points.
c
         if(ngp.eq.4)wt=wt4(ii)
         if(ngp.eq.5)wt=wt5(ii)
         if(ngp.eq.7)wt=wt7(ii)
         do 70 jj=1,11
         do 70 kk=1,11
         do 70 ll=1,7
         stif(jj,kk)=stif(jj,kk)+wt*width*dsftr(jj,ll)*pktd(ll,kk)
         eln(jj,kk)=eln(jj,kk)+wt*width*dsftr(jj,ll)*pknd(ll,kk)
   70    continue
  100    continue
c        write(6,1010)ielem
c        write(6,1005)
c        do 900 ii=1,11
c900     write(6,1000) (stif(ii,jj),jj=1,11)
 1000 format(9(2x,d12.5))
 1005 format(/,'stif')
 1010 format(i4)
         return
         end
c
c
c
         subroutine shape(eta,ielem,aa)
c
         implicit double precision (a-h,o-z)
c
c
         common/shp/dsf(7,11)
c
c
         common/input/tol,table(250),delem(250),vbound(2500),distld,
      .    vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
      .    rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
      .    nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
      .    nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
      .    theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
c
c        initialize shape function matrix
c
         do 10 ii=1,7
         do 10 jj=1,11
   10 dsf(ii,jj)=0.0
```

```
c
      aa=(coord(ielem+1)-coord(ielem))*0.5
c
c       enter values into dsf
c       these include jacobian terms
c
c
c       Q1,Q3,Q2 and derivatives
c
      dsf(1,1)=0.5*(eta**2-eta)
      dsf(1,6)=1.0-eta**2
      dsf(1,7)=0.5*(eta**2+eta)
      dsf(2,1)=(eta-0.5)/aa
      dsf(2,6)=-2.0*eta/aa
      dsf(2,7)=(eta+0.5)/aa
      dsf(3,4)=0.25*(2.0-3.0*eta+eta**3)
      dsf(3,5)=0.25*aa*(1.0-eta-eta**2+eta**3)
      dsf(3,10)=0.25*(2.0+3.0*eta-eta**3)
      dsf(3,11)=0.25*aa*(-1.0-eta+eta**2+eta**3)
      dsf(4,4)=0.25*(-3.0+3.0*eta**2)/aa
      dsf(4,5)=0.25*(-1.0-2.0*eta+3.0*eta**2)
      dsf(4,10)=0.25*(3.0-3.0*eta**2)/aa
      dsf(4,11)=0.25*(-1.0+2.0*eta+3.0*eta**2)
      dsf(5,4)=0.25*6.0*eta/aa**2
      dsf(5,5)=0.25*(-2.0+6.0*eta)/aa
      dsf(5,10)=-0.25*6.0*eta/aa**2
      dsf(5,11)=0.25*(2.0+6.0*eta)/aa
      dsf(6,2)=0.25*(2.0-3.0*eta+eta**3)
      dsf(6,3)=0.25*aa*(1.0-eta-eta**2+eta**3)
      dsf(6,8)=0.25*(2.0+3.0*eta-eta**3)
      dsf(6,9)=0.25*aa*(-1.0-eta+eta**2+eta**3)
      dsf(7,2)=0.25*(-3.0+3.0*eta**2)/aa
      dsf(7,3)=0.25*(-1.0-2.0*eta+3.0*eta**2)
      dsf(7,8)=0.25*(3.0-3.0*eta**2)/aa
      dsf(7,9)=0.25*(-1.0+2.0*eta+3.0*eta**2)
c
c       temporary print
c
c      do 100 ii=1,7
c 100 write(6,1000) (dsf(ii,jj),jj=1,11)
c 1000        format (9(2x,d12.5))
      return
      end
c
c
c
      subroutine postpr(icount,ncount,kcall,ndof,pname,sname)
c
      implicit double precision (a-h,o-z)
c
      character*64 gname,fname
c
c
      common/chac/gname,fname
c
      common/elas/ae,de,fe,he,ej,el,re,te,as,ds,fs
c
      common/input/tol,table(250),delem(250),vbound(2500),distld,
     .  vconc(2500),ey,enu,ht,e1,e2,g12,enu12,enu21,g13,g23,pthick,
     .  rad,linear,isotro,isarch,ishape,inctyp,ninc,imax,
```

```
     .    nelem,nbndry,nbound(250,6),ldtyp,nconc,iconc(2500),
     .    nplies,nforc,iforc(2500),nstres,istres(250),ibndry(2500),
     .    theta(20),idload(250),coord(251),width,nnod,pincr,eiter,ttpi
c
        common/stf/stif(11,11),elp(11),eln(11,11),eld(11)
c
        common/proc/gstif(2500,11),gn(2500,11),gf(2500),gd(2500),vperm(250
0),
     .          vpres(2500)
c
        dimension vforc(2500),xcoord(251),ycoord(251)
        pi=3.14159
c
c       if ishape=1 print global x,y coords to file bshape
c       if ishape=2 figure out symmetric coords as well
c
          if (isarch.eq.0) then
            go to 423
          end if
        if(ishape.eq.1.or.ishape.eq.2.and.ncount.eq.1.and.isarch.eq.1)then
             open(8,file=sname,status='new')
             do 1 ii=1,nnod
             if(ishape.eq.2)then
             xcoord(nnod-1+ii)=rad*cos(pi/2.0-coord(ii)/rad)
             xcoord(nnod+1-ii)=-rad*cos(pi/2.0-coord(ii)/rad)
             ycoord(nnod-1+ii)=rad*sin(pi/2.0-coord(ii)/rad)
             ycoord(nnod+1-ii)=ycoord(nnod-1+ii)
             else
             xcoord(ii)=-rad*cos(pi/2.0+coord(ii)/rad-
coord(nnod)/(2*rad))
             ycoord(ii)=rad*sin(pi/2.0+coord(ii)/rad-coord(nnod)/(2*rad))
             endif
    1        continue
             do 100 ii=1,nnod
  100        write(8,2000)xcoord(ii),ycoord(ii)
             if(ishape.eq.2)then
             do 110 ii=2,nnod
  110        write(8,2000)xcoord(nnod+ii-1),ycoord(nnod+ii-1)
             endif
             write(8,2010)
        endif
c
c       global displacements for straight beams
c
  423   if(ishape.eq.1.and.ncount.eq.1.and.isarch.eq.0)then
             open(8,file=sname,status='new')
             do 2 ii=1,nnod
             xcoord(ii)= coord(nnod)-coord(ii)
             ycoord(ii)=0.0
    2        write(8,2000)xcoord(ii),ycoord(ii)
             write(8,2010)
        endif
c
c       print out global displacements
c
        write(6,1000)
        write(6,1010)
        write(6,1020)ncount,icount
        write(6,1030)
        do 90 ii=0,nnod-1
```

```
c
c       x,y for arches
c
        if(ishape.eq.1.and.isarch.eq.1)then
        if(ishape.eq.1.and.isarch.eq.1.and.ii.eq.0)write(8,2020)ncount
             xcoord(ii+1)=-(rad-gd(6*ii+4))*
        .        cos(pi/2.0+coord(ii+1)/rad-coord(nnod)/(2*rad)+
        .           gd(6*ii+1)/rad)
             ycoord(ii+1)=(rad-gd(6*ii+4))*
        .        sin(pi/2.0+coord(ii+1)/rad-coord(nnod)/(2*rad)+
        .           gd(6*ii+1)/rad)
             write(8,2000)xcoord(ii+1),ycoord(ii+1)
        endif
        if(ishape.eq.2.and.isarch.eq.1)then
        if(ii.eq.0)write(8,2020)ncount
             xcoord(nnod+ii)=(rad-gd(6*ii+4))*
        .           cos(pi/2.0-coord(ii+1)/rad +gd(6*ii+1)/rad)
             xcoord(nnod-ii)=-xcoord(nnod+ii)
             ycoord(nnod+ii)=(rad-gd(6*ii+4))*
        .           sin(pi/2.0-coord(ii+1)/rad+gd(6*ii+1)/rad)
             ycoord(nnod-ii)=ycoord(nnod+ii)
        endif
c
c       x,y for straight beams
c
        if(ishape.eq.1.and.isarch.eq.0)then
        if(ishape.eq.1.and.isarch.eq.0.and.ii.eq.0)write(8,2020)ncount
             xcoord(ii+1)=coord(nnod)-coord(ii+1)-gd(6*ii+1)
             ycoord(ii+1)=-gd(6*ii+4)
             write(8,2000)xcoord(ii+1),ycoord(ii+1)
        endif
        write(6,1040)ii+1,(gd(6*ii+jj),jj=1,5)
   90   write(6,1050)gd(6*ii+6)
        if(ishape.eq.2.and.isarch.eq.1)then
             do 95 ii=1,2*nnod-1
   95        write(8,2000)xcoord(ii),ycoord(ii)
        endif
        if(ishape.ge.1)write(8,2010)
c
c       compute equivalent forces requested
c
   3    do 5 ii=1,ndof
        gf(ii)=0.0d0
        do 5 jj=1,11
        gstif(ii,jj)=0.0d0
   5    gn(ii,jj)=0.0d0
c
c       loop over all elements for stiffness and forces
c
   9    do 30 ielem=1,nelem
        do 10 ii=1,11
   10   eld(ii)=gd(ii+(ielem-1)*6)
c
        call stiff(ielem,icount,ncount,kcall)
c
c       Assemble global stiffness array, gstif, global equilibrium
c       stiffness, gn, in banded form. Half-bandwidth=11.  Also
c       assemble global force vector, gf.
c
        nr=(ielem-1)*6 + 1
```

```fortran
      do 30 jj=0,10
      gf(nr+jj)=gf(nr+jj)+elp(jj+1)
      do 30 kk=1,11-jj
      gstif(nr+jj,kk)=gstif(nr+jj,kk)+stif(jj+1,kk+jj)
      if(linear.eq.1)goto 30
      gn(nr+jj,kk)=gn(nr+jj,kk)+eln(jj+1,kk+jj)
   30 continue
c
c     calculate the residual force vector for nonlinear
c     analysis. -[gn]*{gd}+R=-[k+n1/2+n2/3]*{q}+R=gf
c
          do 60 jj=1,nforc
      ii=iforc(jj)
          add=0.
          do 50 kk=1,ii-1
          if(ii-kk+1 .gt. 11)goto 50
          add=add+gn(kk,ii-kk+1)*gd(kk)
   50     continue
          res=0.
          do 55 ll=1,11
          if(ll+ii-1 .gt. ndof)goto 55
          res=res + gn(ii,ll)*gd(ll+ii-1)
   55     continue
c
c     compute nodal force
c
      vforc(jj)=res+add
   60 continue
c
c     print nodal forces and create plot file
c
      open(7,file=pname,status='new')
      if(ncount.eq.1)write(7,1065)0.0,0.0,0.0
      write(6,1060)
      do 70 ii=1,nforc
      write(7,1065)gd(iforc(ii)),gd(iforc(ii)-3),vforc(ii)
   70 write(6,1070)iforc(ii),vforc(ii)
 1000 format(/)
 1010 format(1x,'Results of nonlinear analysis')
 1020 format(1x,'increment=',i3,'  iteration=',i3)
 1030 format(1x,'Node',7x,'V',12x,'Psi-s',9x,'Psi-ss',9x,'W',9x,'W-s')
 1040 format(1x,i4,5(2x,d12.5))
 1050 format(1x,'Midnode v:',3x,d12.5)
 1060 format(1x,/,'Equivalent nodal forces:')
 1065 format(1x,d12.5,2x,d12.5,2x,d12.5)
 1070 format(1x,'DOF no:',i4,2x,'Force:',1x,d12.5)
 2000 format(1x,f12.5,2x,f12.5)
 2010       format(//)
 2020 format(/,1x,i4)
      return
      end
```

Bibliography

1.   Belytschko, Ted and Lawrence W. Glaum. "Application of Higher Order Corotational Stretch Theories to Nonlinear Finite Element Analysis," Computers and Structures 10 (1979): 175-182.

2.   Cook, Robert D., David S. Malkus, and Michael E Plesha. Concepts and Applications of Finite Element Analysis. New York: John Wiley & Sons, 1989.

3.   Creaghan, Stephen G. Nonlinear Large Displacement and Moderate Rotational Characteristics of Composite Beams Incorporating Transverse Shear Strain. MS thesis, AFIT/GA/ENY/92D-10. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1992 (AD-A258908).

4.   Creaghan, S. G. and A. N. Palazotto. "Nonlinear Large Displacement and Moderate Rotational Characteristics of Composite Beams Incorporating Transverse Shear Strain," Computers and Structures 4 (1994): 357-371.

5.   DeDeppo, D.A. and R Schmidt. "Large Deflections and Stability of Hingless Circular Arches Under Interacting Loads," Journal of Applied Mechanics 41 (1974): 989-994.

6.   -----. "Instability of Clamped-Hinged Circular Arches Subjected to a Point Load," Journal of Applied Mechanics 42 (1975): 894-896.

7.   Epstein, Marcelo and David W. Murray. "Large Deformation In-Plane Analysis of Elastic Beams," Computers and Structures 6 (1976): 1-9.

8.   Fung, Y.C. and Kaplan, A. "Buckling of Low Arches and Curved Beams of Small Curvature," NACA TN 2840 (1952).

9.   Gallagher, R. H. Finite Element Analysis: Fundamantals. Enlgewood Cliffs: Prentice-Hall Inc., 1975.

10.   Huang, Nai-Chien. "Unsymmetrical Buckling of Thin Shallow Spherical Shells," Journal of Applied Mechanics 31 (1964): 447-457.

11.   Huddelston, J.V. "Finite Deflections and Snap-Through of High Circular Arches," Journal of Applied Mechanics 31 (1968): 763-769.

12.   Kaplan, Abner, "Buckling of Spherical Shells." Thin-Shell Structures: Theory, Experiment, and Design. Ed. Y. C. Fung and E. E. Sechler. Englewood Cliffs: Prentice-Hall Inc., 1974.

13.   Miller, Daniel A. Nonlinear Large Deformation Theory of Composite Arches Using Truncated Rotations. MS thesis, AFIT/GAE/ENY/93D-22. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, December 1993 (AD-A273726).

14.   Mindlin, R.D. "Influence of Rotary Inertia and Shear on Flexural Motions of Isotropic, Elastic Plates," <u>Journal of Applied Mechanics</u> 18 (1951): 31-38.

15.   Minguet, Pierre and John Dugundji. "Experiments and Analysis for Composite Blades Under Large Deflections Part I: Static Behavior," <u>AIAA Journal</u> 28 (1990): 1573-1579.

16.   Mondkar, D.P. and G.H. Powell. "Finite Element Analysis of Non-Linear Static and Dynamic Response," <u>International Journal for Numerical Methods in Engineering</u> 11 (1977): 499-520.

17.   Noguchi, H. and T. Hisada. "Sensitivity Analysis in Post-Buckling Problems of Shell Structures," <u>Computers and Structures</u> 47 (1993): 699-710.

18.   Oliver, J. and E. Oñate. "A Total Lagrangian Formulation for the Geometrically Nonlinear Analysis of Structures using Finite Elements. Part II: Arches, Frames, and Axisymmetric Shells," <u>International Journal for Numerical Methods in Engineering</u> 23 (1986): 253-274.

19.   Palazotto, A. N. and S. T. Dennis. <u>Nonlinear Analysis of Shell Structures</u>. Washington D.C.: American Institute of Aeronautics and Astronautics, 1992.

20.   Reddy, J.N. "A Simple Higher-Order Theory for Laminated Composite Plates," <u>Journal of Applied Mechanics</u> 51 (1984): 745-752.

21.   Reissner, Eric. "The Effect of Transverse Shear Deformation on the Bending of Elastic Plates," <u>Journal of Applied Mechanics</u> 12 (1945): A69-A77.

22.   Saada, Adel S. <u>Elasticity Theory and Applications</u>. Malabar: Krieger, 1989.

23.   Sabir, A.B. and A.C. Lock. "Large Deflexion, Geometrically Non-Linear Finite Element Analysis of Circular Arches," <u>International Journal of Mechanical Sciences</u> 15 (1973): 37.47.

24.   Schmidt, R. and D. A. DaDeppo. "Nonlinear Theory of Arches with Transverse Shear Deformation and Rotary Inertia," <u>Industrial Mathematics</u> 21 (1971): 33-49.

25.   Smith, R. A. <u>Higher-Order Thickness Expansions for Cylindrical Shells</u>. PhD Dissertation. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, 1991 (AD-242017) .

26.   Smith, R.A. and A. N. Palazotto. "Comparison of Eight Variations of a Higher-Order Theory for Cylindrical Shells," <u>AIAA Journal</u> 31 (1993): 1125-1132.

27.   Surana, Karan S. "Geometrically Nonlinear Formulation for the Curved Shell Elements," <u>International Journal for Numerical Methods in Engineering</u> 19 (1983): 581-615.

28.   Tsai, C.T. and A.N. Palazotto.   "Nonlinear and multiple Snapping
       Responses of Cylindrical Panels Comparing Displacement Control and
       Rik's Method," <u>Computers and Structures</u> 41 (1991): 605-610.

29.   Zienkiewicz, O. C. and R. L. Taylor.   <u>The Finite Element Method,</u>
       4th ed.   New York: McGraw Hill, 1991.

Vita

Captain John C. Bailey was born on January 27, 1969 in Honolulu, Hawaii. He graduated from Palm Springs High School in Palm Springs, California in June 1986. He then attended the United States Air Force Academy where he graduated with a Bachelor of Science in Engineering Mechanics in May 1990. Upon graduation, he received a regular commission in the USAF and spent his first duty assignment at the 384th Bomb Wing, McConnell Air Force Base, Kansas. He was assigned to the Air Force Institute of Technology in May 1993.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE December 1994	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE INVESTIGATION INTO THE BEHAVIOR OF GEOMETRICALLY NONLINEAR COMPOSITE ARCHES	5. FUNDING NUMBERS

6. AUTHOR(S)

John C. Bailey, Captain, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, WPAFB OH 45433-6583	8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GAE/ENY/94D-02

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Dr. Arnold Mayer WL/FIV WPAFB OH 45433 (513) 255-3311	10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This research modifies the existing finite element formulation of a potential energy based large deformation and moderate rotation theory. Hermitian shape functions replace the existing linear bending angle interpolations. Negligible differences between the two formulations indicate the underlying kinematics limit the accuracy, not the finite element interpolations. Using the new program, numerous nonlinear arch geometries are modeled to investigate the effects of arc length and thickness variations. Local and global snapping phenomena are captured as well as through the thickness shear driven nonlinearities.

14. SUBJECT TERMS Large rotation kinematics. Composite arches. Parabolic shear. Nonlinear finite element. Total Lagrangian			15. NUMBER OF PAGES 97
			16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL